Práctica 2 Competición Titanic en kaggle

Javier Maestre Deusto y Miguel López Marzabal

10 de junio, 2020

Contents

1. Descripción del dataset. ¿Por qué es importante y qué pregunta/problema pretende responder?

Se trata de un dataset compuesto por los pasajeros del Titanic y cual fue su desenlace, si sobrevivieron o no. La finalidad del estudio es predecir cual fué el destino final de un conjunto de pasajeros en los cuales no hay datos del desenlace. Este estudio forma parte de una competición abierta en el portal de ciencia de datos Kaggle, por lo que está ampliemente documentado y existe una gran diversidad de métodos por los cuales se puede obtener el resultado. La competición está basada en el porcentaje de acierto de la predicción hecha.

El dataset está compuesto por los siguientes campos:

Variable	Definition	Key
survival	Survival	0 = No, 1 = Yes
pclass	Ticket class	1 = 1st, 2 = 2nd, 3 = 3rd
sex	Sex	
Age	Age in years	
sibsp	# of siblings / spouses aboard the Titanic	
parch	# of parents / children aboard the Titanic	
ticket	Ticket number	
fare	Passenger fare	
cabin	Cabin number	
embarked	Port of Embarkation	C = Cherbourg, Q = Queenstown, S = Southampton

** Variable Notes **

pclass: A proxy for socio-economic status (SES) 1st = Upper 2nd = Middle 3rd = Lower

age: Age is fractional if less than 1. If the age is estimated, is it in the form of xx.5

sibsp: The dataset defines family relations in this way... Sibling = brother, sister, stepbrother, stepsister Spouse = husband, wife (mistresses and fiancés were ignored)

parch: The dataset defines family relations in this way... Parent = mother, father Child = daughter, son, stepdaughter, stepson Some children travelled only with a nanny, therefore parch=0 for them.

2. Integración y selección de los datos de interés a analizar.

En este apartado, a parte de la integración y selección de datos, vamos a realizar conversión(construcción de atributos, agregación, normalización, discretización).

Unimos los conjuntos de train y test para analizar los datos al completo

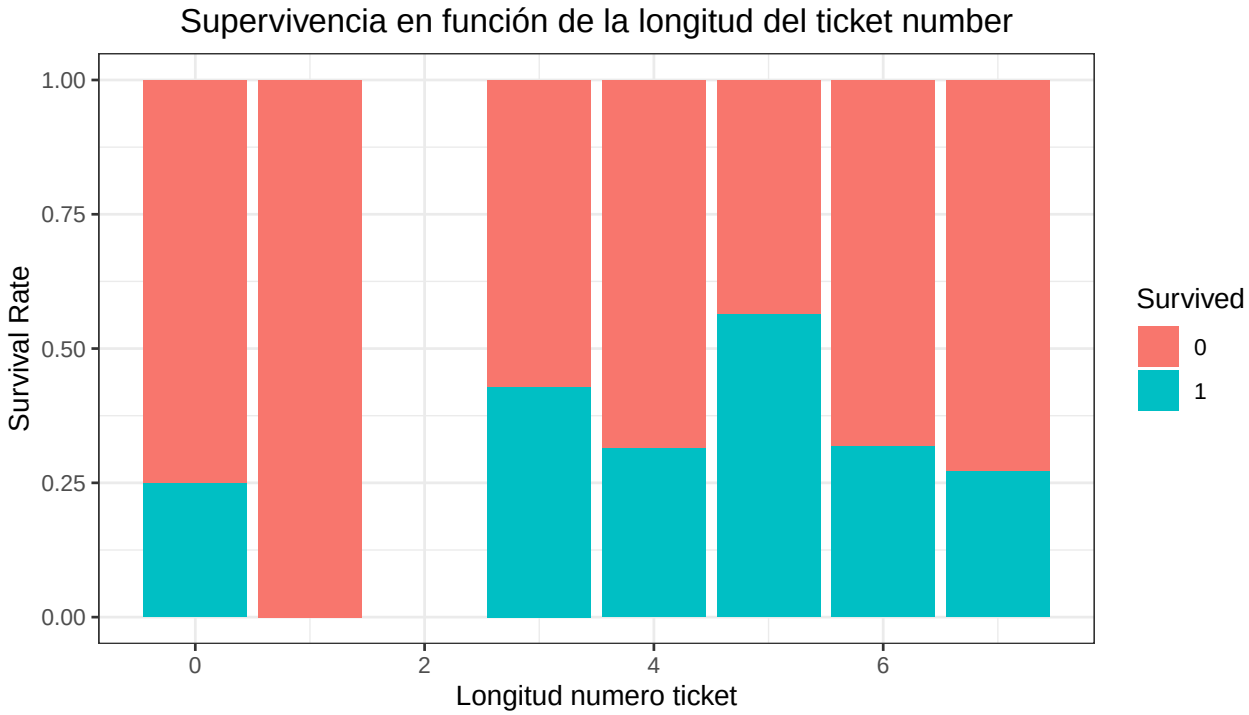
Trabajo con las diferentes variables

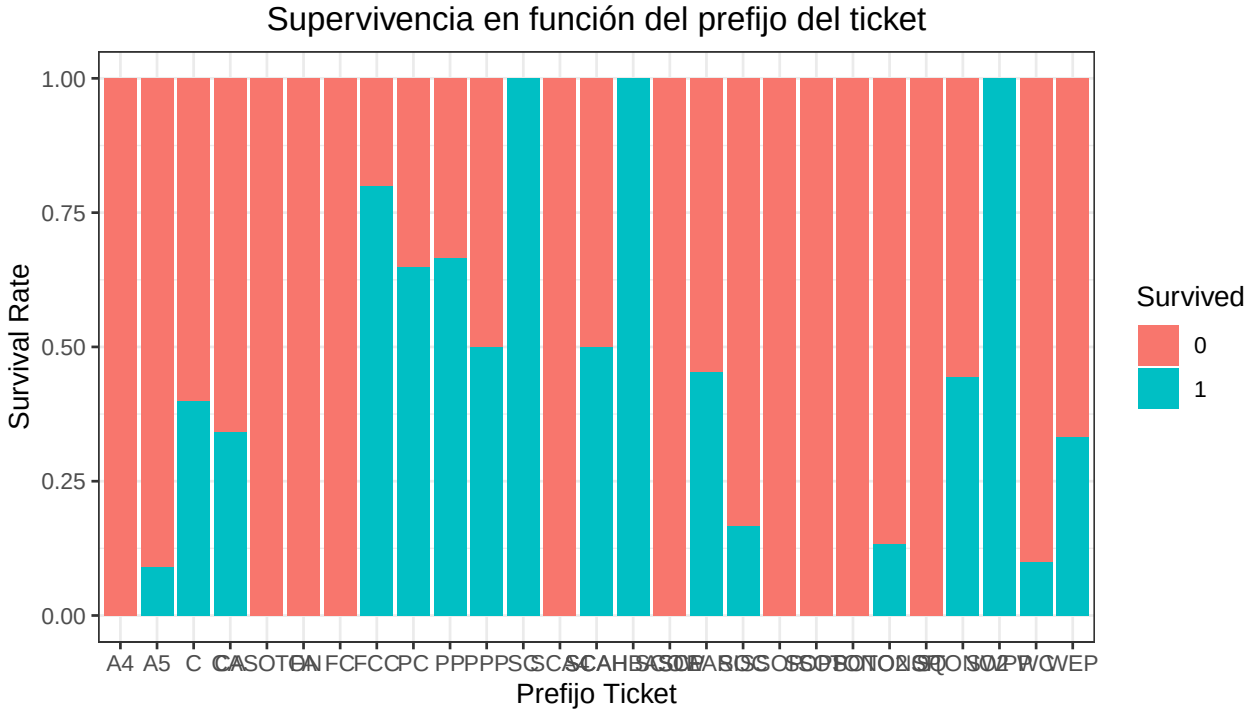
En este documento vamos a separar el trabajo hecho en cada variable, por tanto irán mezcladas cosas de preprocesado

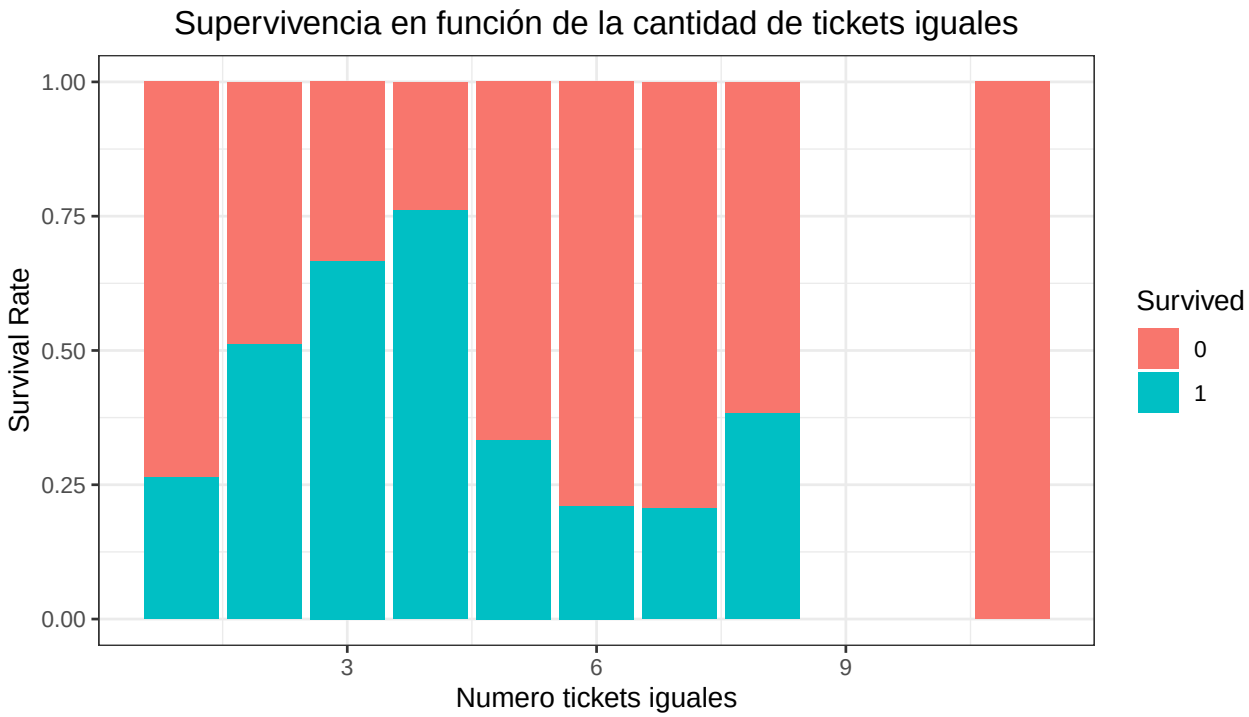
Trabajo con Ticket

Partimos el campo ticket en prefijo y número de ticket. A partir del numero de ticket obtenemos grupos en función de la cantidad de numero del número de ticket y vemos como se relaciona con sobrevivir. Vemos

como se relacionan los prefijos del ticket con sobrevivir. Obtenemos una variable que nos indica la catidad de números de tickets(TicketFreq) iguales y vemos su relacion con sobrevivir. Hemos tenido que corregir 4 casos de valores perdidos en nuestra nueva variable debido a que estos casos no tenian número de ticket.



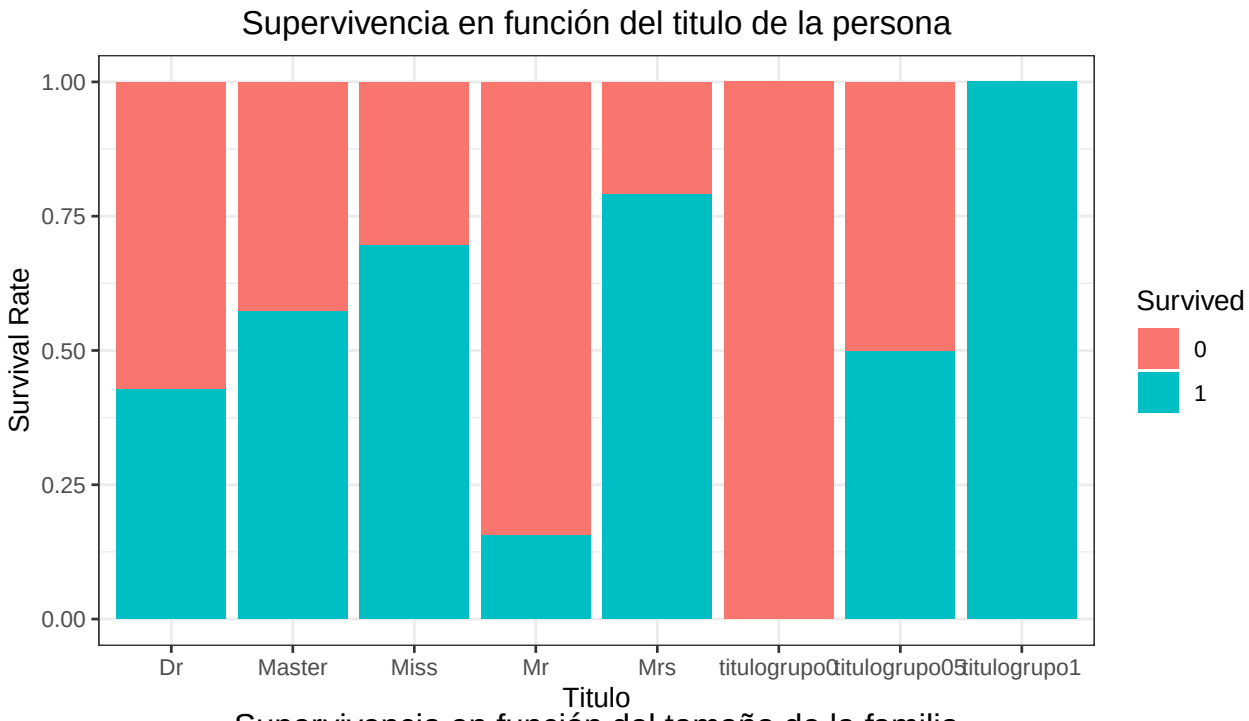


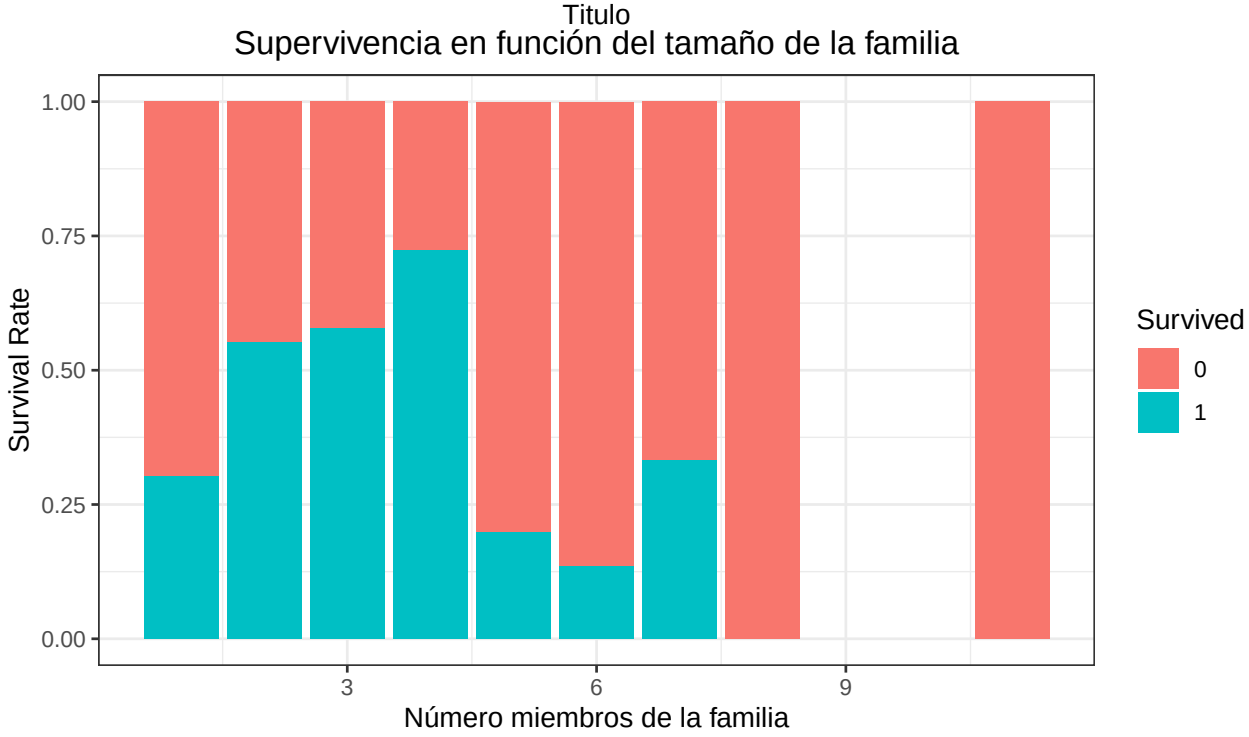


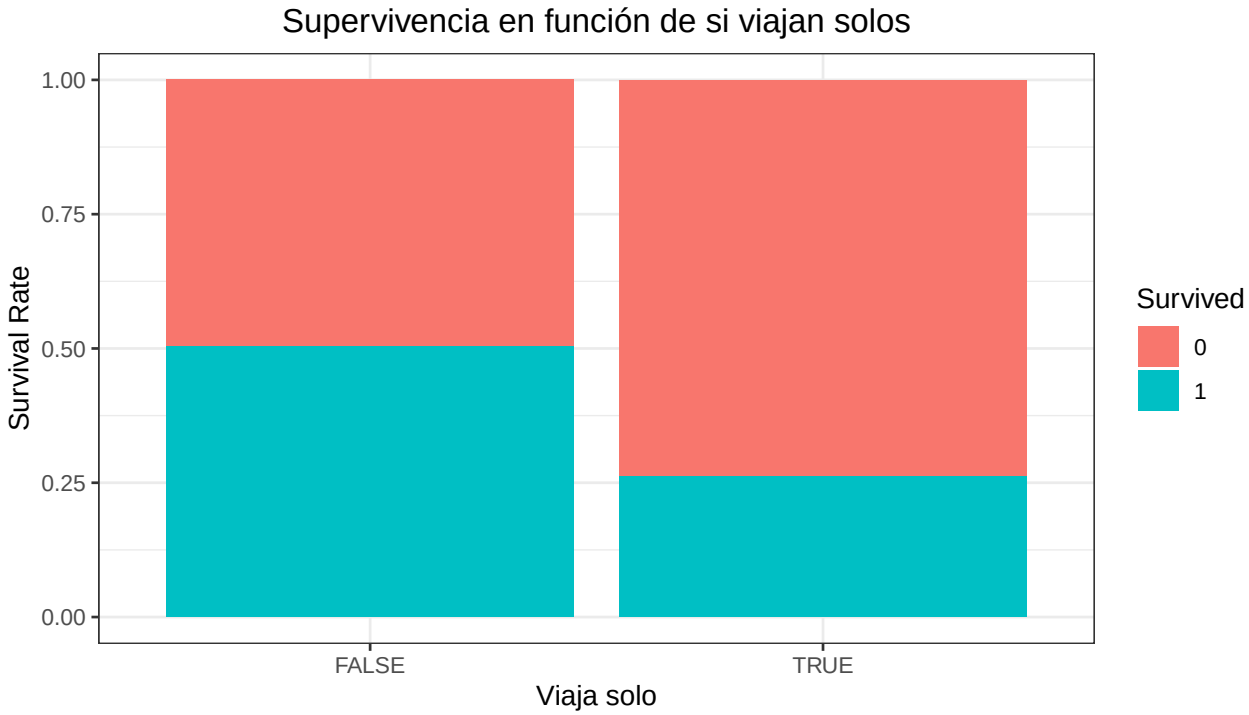
Trabajo con Name

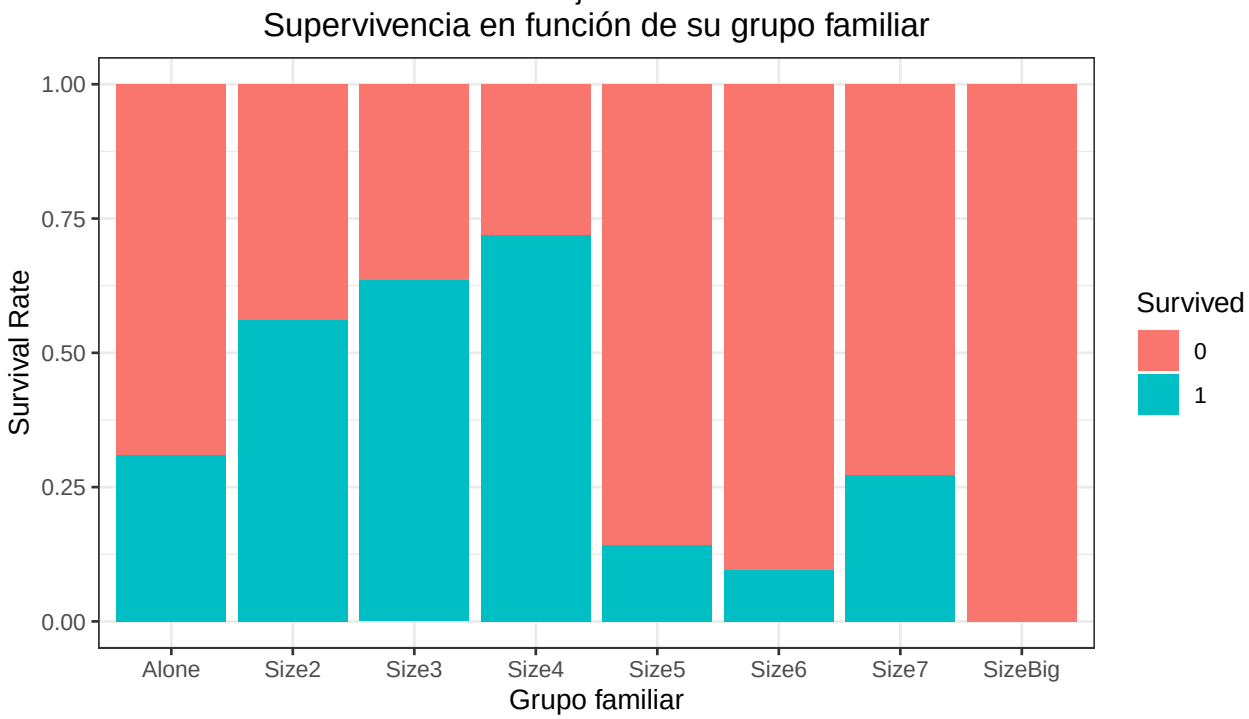
Partimos el campo name para obtener por un lado el Apellido del pasajero y por otro lado el título que recibe esa persona.

1. Agrupamos los titulos poco frecuentes en tres grupos según su probabilidad de supervivencia, los que todas las personas con ese título han muerto van a 'titulogrupo0', si han sobrevivido todos a 'titulogrupo1' y si la probabilidad es diferente al grupo 'titulogrupo05'. Comprovamos que la relación de los grupos con supervivencia.
2. Creamos una nueva variable llamada FamSize que nos indica el número de familiares a bordo, esto se obtiene con 'SibSp + Parch + 1'. Mostramos gáficamente la relación con supervivencia.
3. Creamos una nueva variable 'LoneWolfs' que nos indica si una persona viaja totalmente sola, eso lo obtenemos con los elementos que cumplen esta condición 'FamSize == 1 & TicketFreq==1' y visualizamos la relación de la nueva variable con sobrevivir.
4. Vamos a agrupar las familias para ver exactamente cuantos familiares viajaban juntos. Esto lo haremos concatenando el Apellido con el número de familiars y el número de ticket. Hemos añadido el número de ticket porque si solo juntamos apellido y número de familiares se mezclaban familias que coincidían en apellido y numero de familiares pero viajaban con diferente ticket. Tras agruparlos correctamente creamos grupos genéricos según el número real de familiares que viajaban juntos. Finalmente visualizamos la relación de nuestra nueva variable 'FamilyIDTKGrouped' con la supervivencia.


Supervivencia en función del titulo de la persona
Survival Rate
1.00
0.75
0.50
0.25
0.00
Dr
Master
Miss
Mr
Mrs
titulogrupo0
titulogrupo05
titulogrupo1
Titulo
Survived
0
1

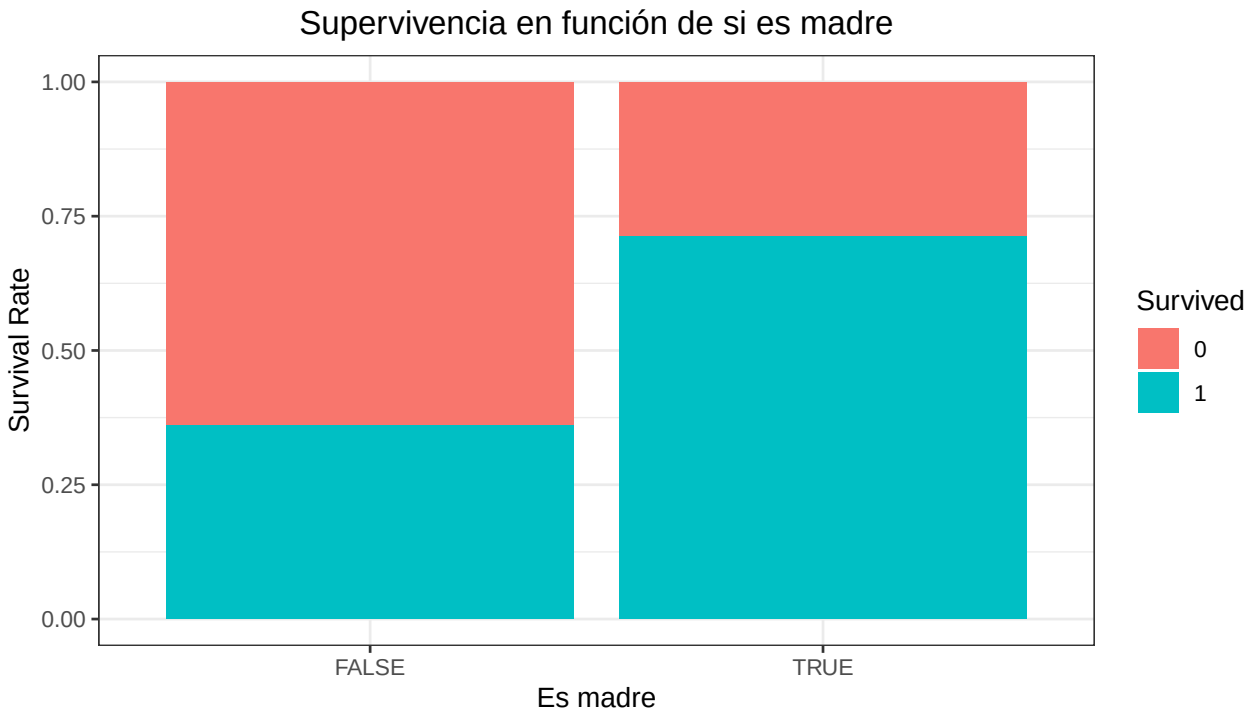

Supervivencia en función del tamaño de la familia
Survival Rate
1.00
0.75
0.50
0.25
0.00
3
6
9
Número miembros de la familia
Survived
0
1





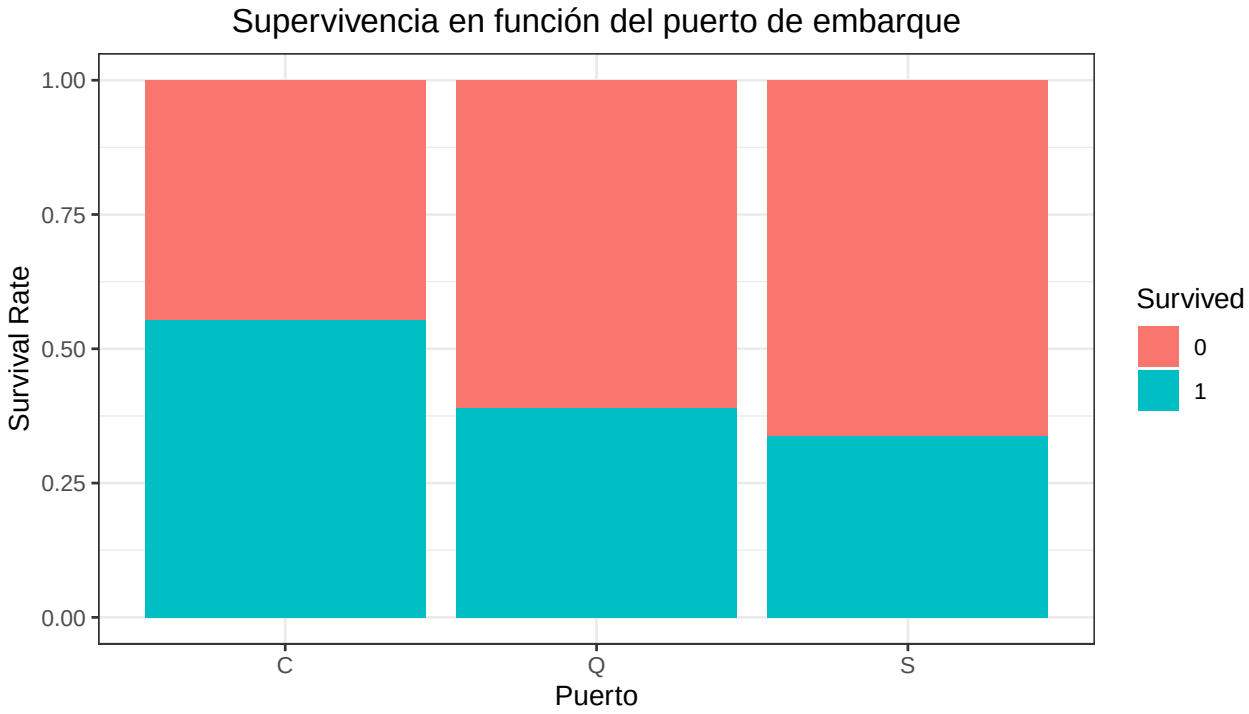
Trabajo con Parch

Creamos una nueva variable que nos indica si la persona es una madre o no. Esta variable se obtiene de las observacviones que cumplen la condicion de tener el título 'Mrs' y que la variable Parch que indica el número de hijos o padres sea mayo que 0. Visualizamos la relación de nuestra nueva variable 'IsMother' con la supervivencia.



Trabajo con Embarqued

En esta variable nos encontramos varios valores perdidos que sustituimos por el valor más frecuente. Visualizamos la relación de Embarqued con la supervivencia.



Trabajo con Cabina

Existen demasiados valores perdidos y no tenemos una forma buena de predecirlos, sin que produzacan demasiado ruido, por lo que vamos a ignorar esta variable.

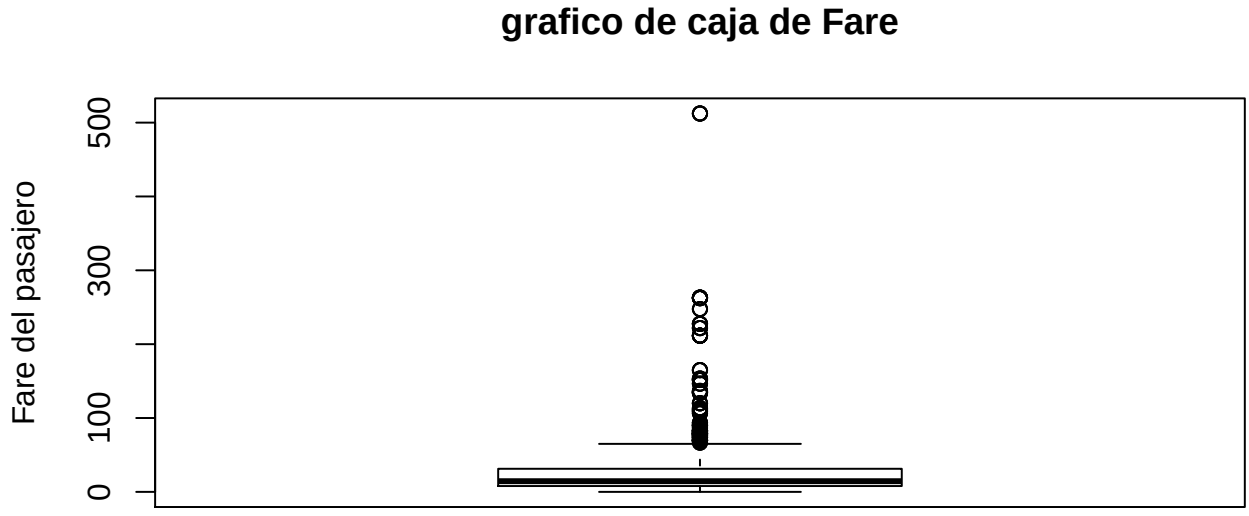
Trabajo con Fare

Existen varios valores perdidos que sustituimos por la mediana del 'Fare' en función de la clase, el sexo y el número de hijos. También existen ciertos pasajeros con Fare 0, pero los vamos a dejar como están.

Vemos el gráfico de caja de la variable y observamos que exiten valores extremos que pueden afectar a la variable para solucionar esto vamos agrupamos Fare en categorías y añadimos los valores extremos a la última categoría. La agrupación se crea en la variable 'FareGroups' y visualizamos la relación de nuestras categorías con la supervivencia.

Otenemos una nueva variable con el Fare individual de cada persona, que obtenemos de dividir el Fare por el número de tickets iguales. Vemos que esta variable también está afectada por valores extremos, por lo que creamos una agrupación como en el caso de Fare(FareIndGroups) y visualizamos la relación de nuestros nuevos grupos con la supervivencia.

Como idea feliz vamos a probar a juntar las variables de Fare y sex, ya que hay diferencias significativas en la supervivencia de hombres y mujeres. Creamos una nueva variable 'FareIndividualBySex' que incrementa el Fare de las mujeres en 60, con esto conseguimos paliar un poco los efectos de los valores extremos, pero tendremos que analizar más adelante si es útil o no. También creamos una variable(FareIndividualBySexGroups) que agrupe estos nuevos Fares y visualizamos la relación de nuestros grupos con la supervivencia.



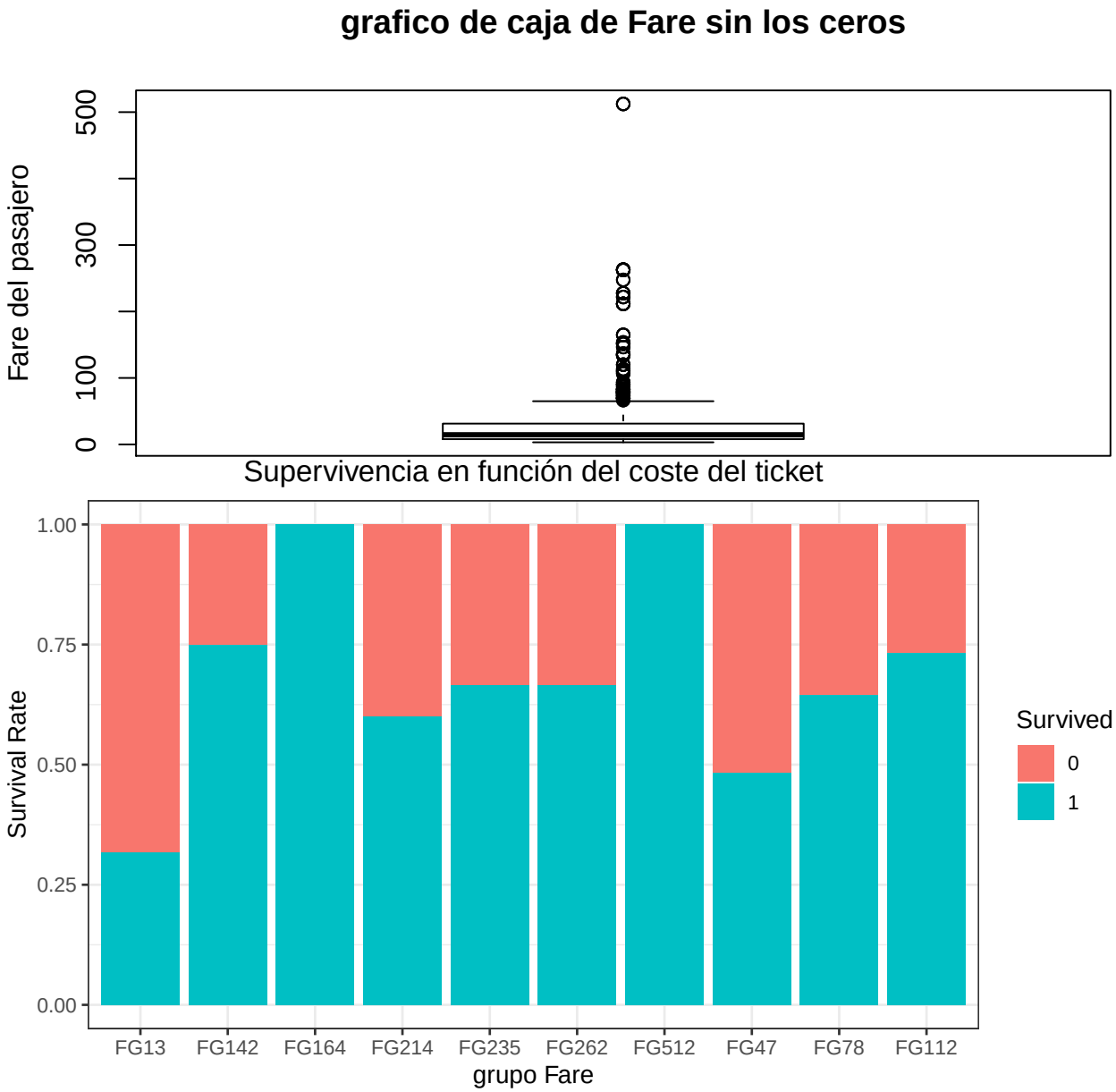


grafico de caja de Fare sin los ceros
Fare del pasajero
0
100
300
500
Supervivencia en función del coste del ticket
Survival Rate
1.00
0.75
0.50
0.25
0.00
FG13
FG142
FG164
FG214
FG235
FG262
FG512
FG47
FG78
FG112
grupo Fare
Survived
0
1

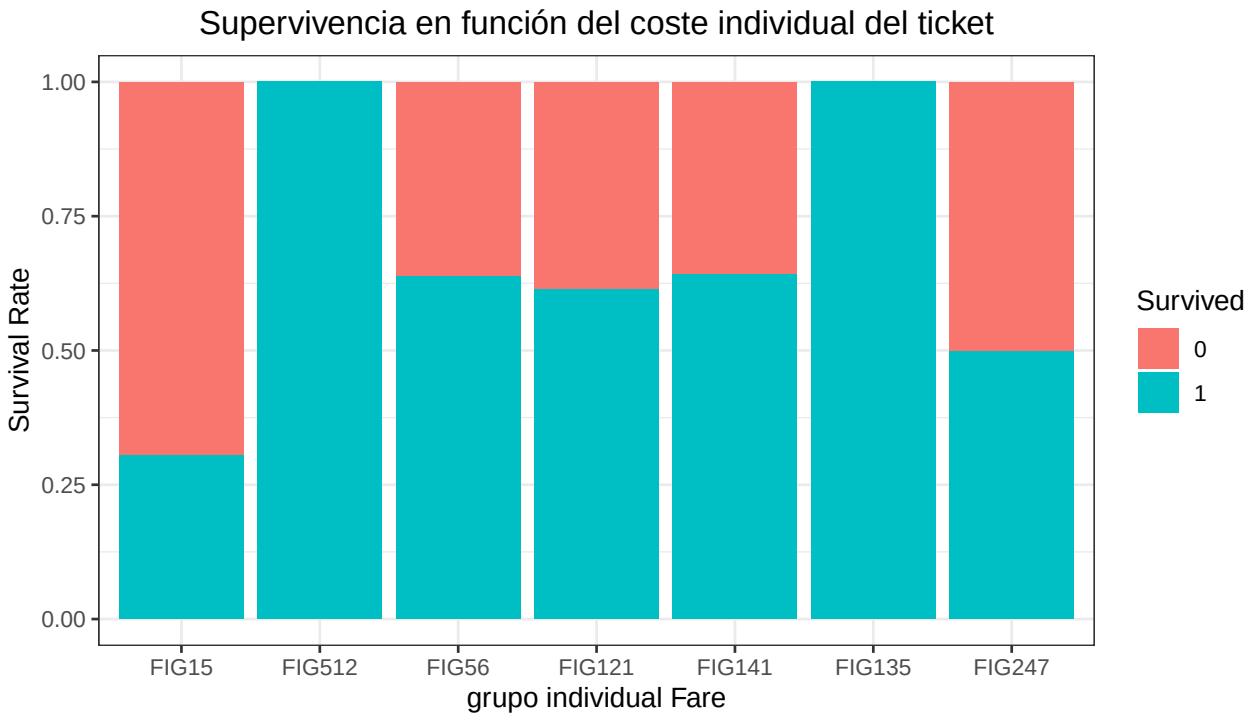
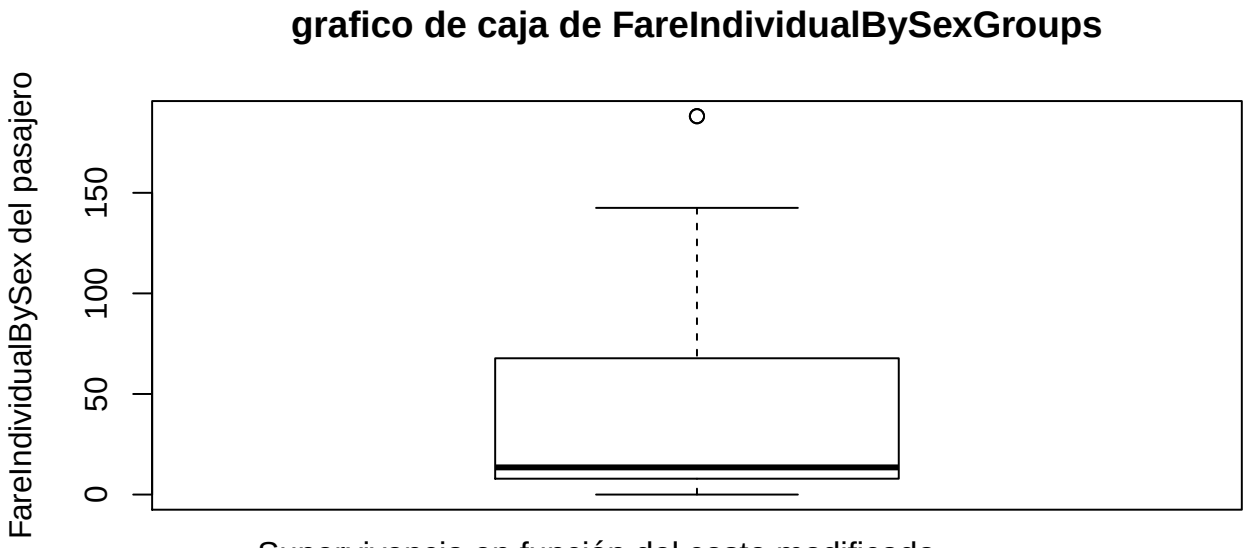
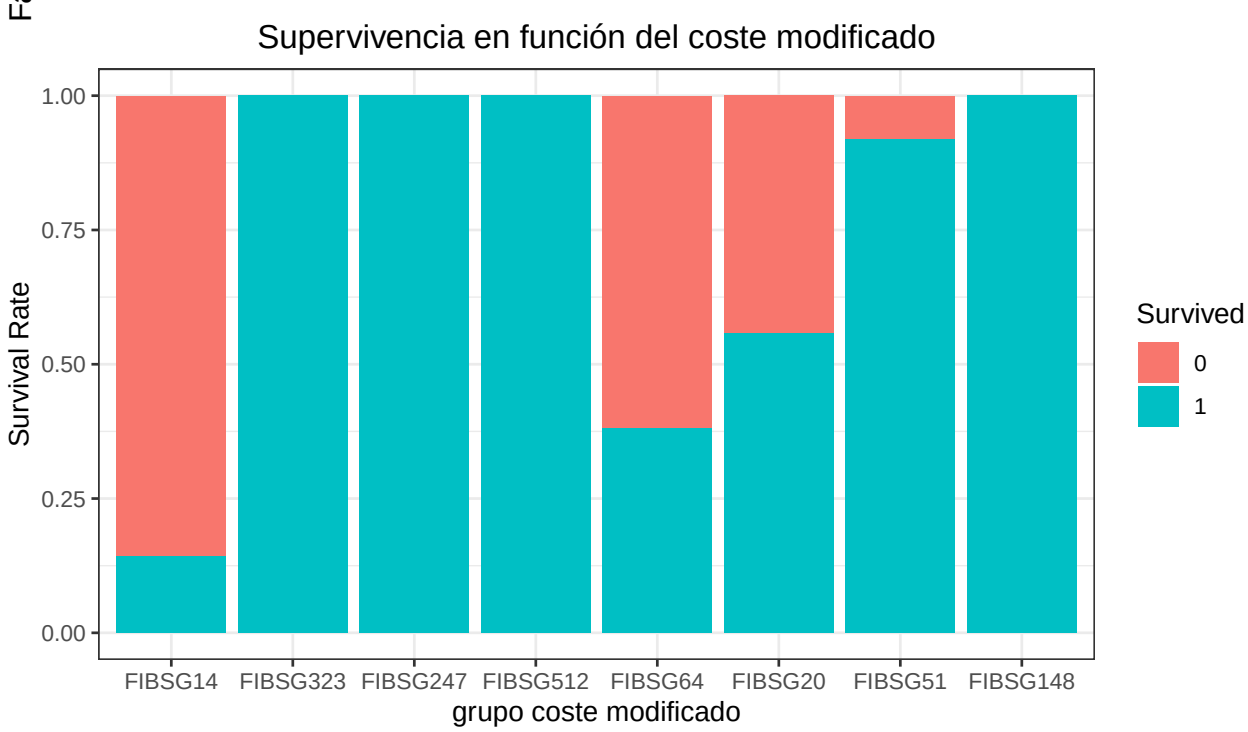

Supervivencia en función del coste individual del ticket
Survival Rate
1.00
0.75
0.50
0.25
0.00
FIG15
FIG512
FIG56
FIG121
FIG141
FIG135
FIG247
grupo individual Fare
Survived
0
1


grafico de caja de FareIndividual
FareIndividual del pasajero
0
20
60
100





Trabajo con Edad

Mostramos el Histograma de la variable y vemos el número de agrupaciónes óptimo para agrupar la variable. Creamos la variable 'ageGroups' con los grupos.

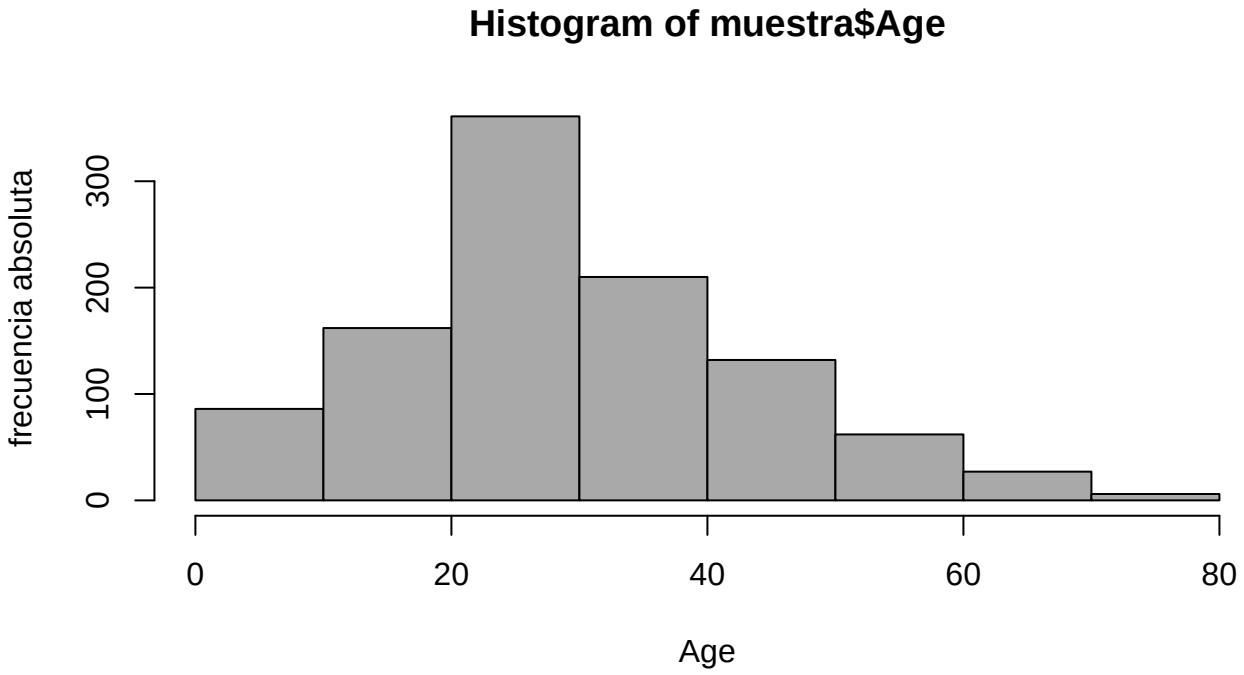
Primero vamos a limpiar los valores perdidos, esto lo vamos a hacer de varias maneras y luego estudiaremos cual es la mejor:

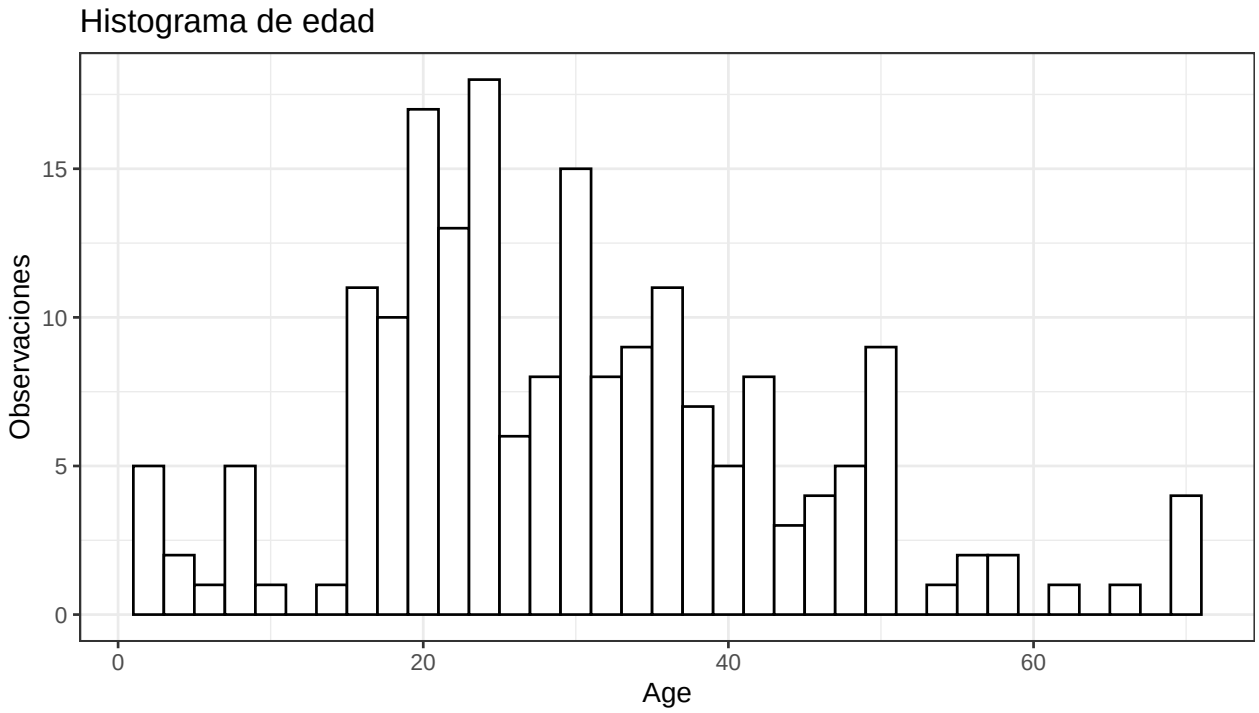
- El primer método de imputación será imputar por la edar media del titulo de la persona. Creamos la variable Ages.
- El segundo método es KNN con el que imputamos los valores perdidos de Age en función de las variables "Pclass", "Sex", "SibSp", "Parch", "Fare", "Embarked", "Titulo", 'FamSize'. Creamos la variable KNNAges

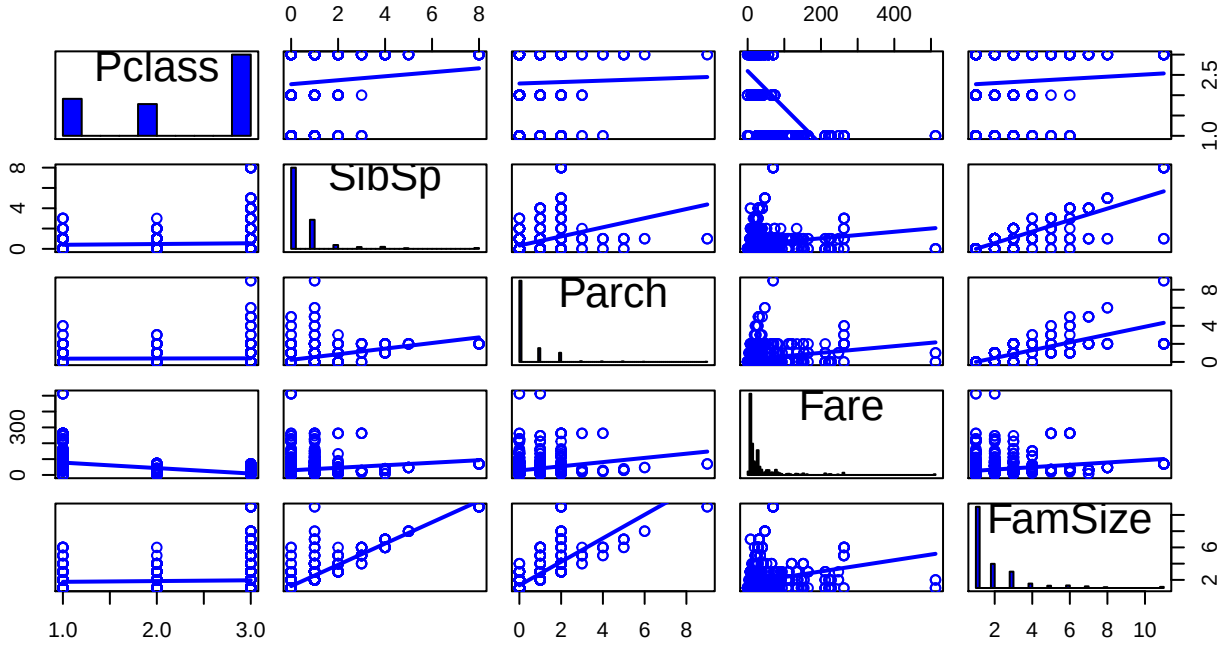
- El tercer método es por regresión. Primero vemos como se ditribuye la edad visualizando su histograma con un mayor número de divisiones. Representamos la relación entre las variables cuantitativas y también mostramos la relación entre las variables cualitativas y la edad. Generamos un modelo para predecir el logaritmo de la edad(esto mejora la falta de normalidad de la variable Age) a partir de las variables Pclass, SibSp, Parch, Fare, Sex, Embarked, IsMother, LoneWolfs, FamSize, Titulo y usamos stepAIC para que seleccione el modelo que mejor funciona, partir de esas variables se va quedando con las variables más representativas. Visualizamos los gráficos de los residuos del modelo y creamos la variable RegresionAges para almacenar las edades imputadas.

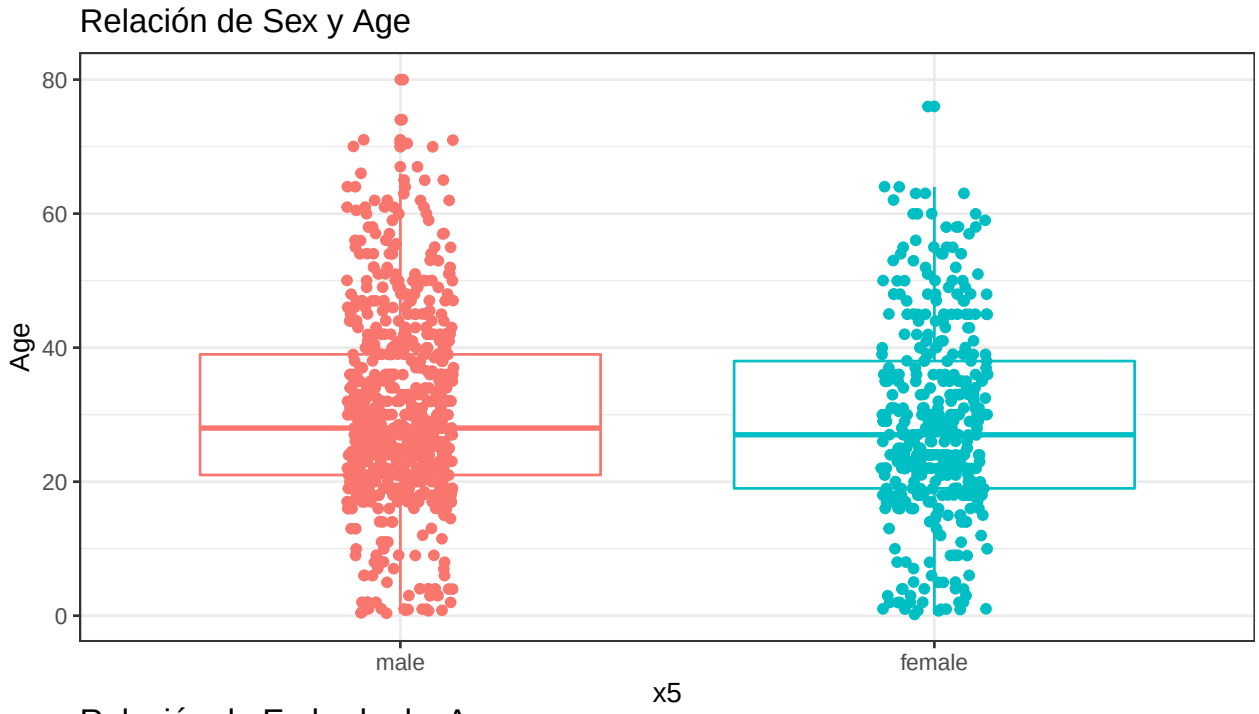
Visualizando el gráfico de caja de la variable observamos que hay algunos Outliers, para solucionarlo agrupamos las edades nuevamente tras la imputación de los valores perdidos y visualizamos la relación con supervivencia.

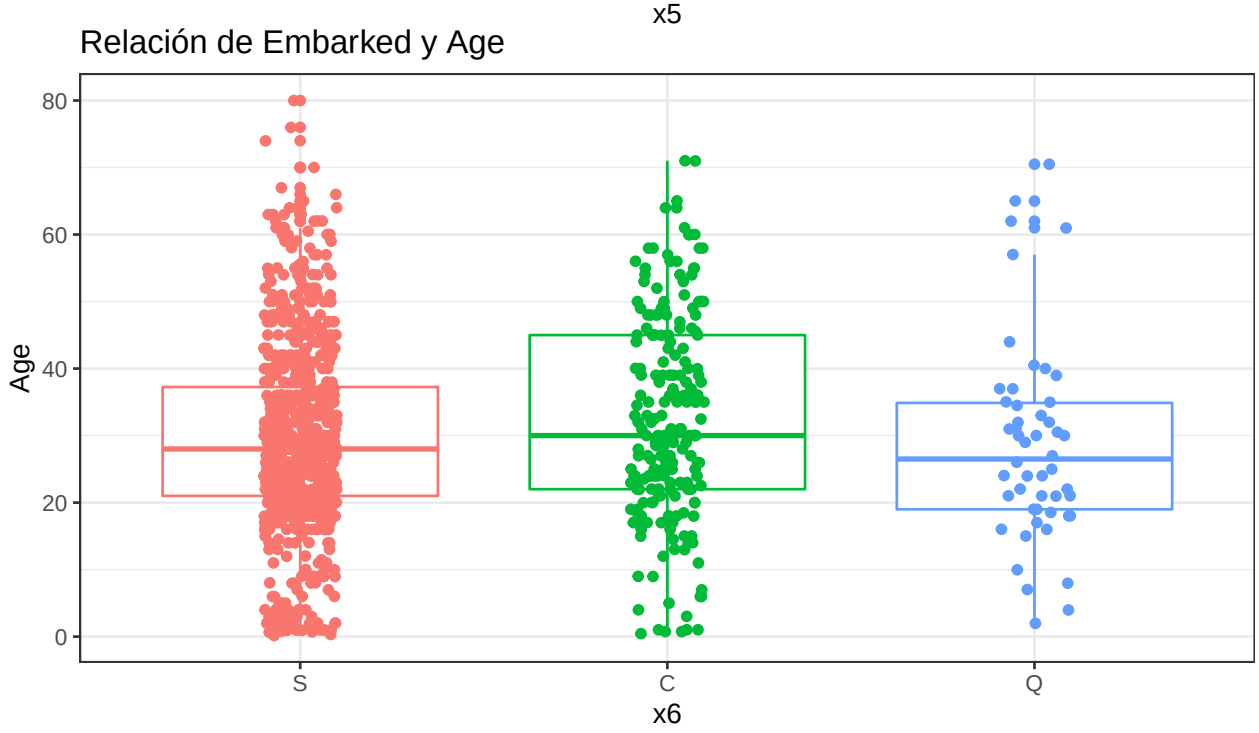
Creamos una nueva variable que nos indica si una persona menor de 16 años está viajando sola y visualizamos la relación con la supervivencia.

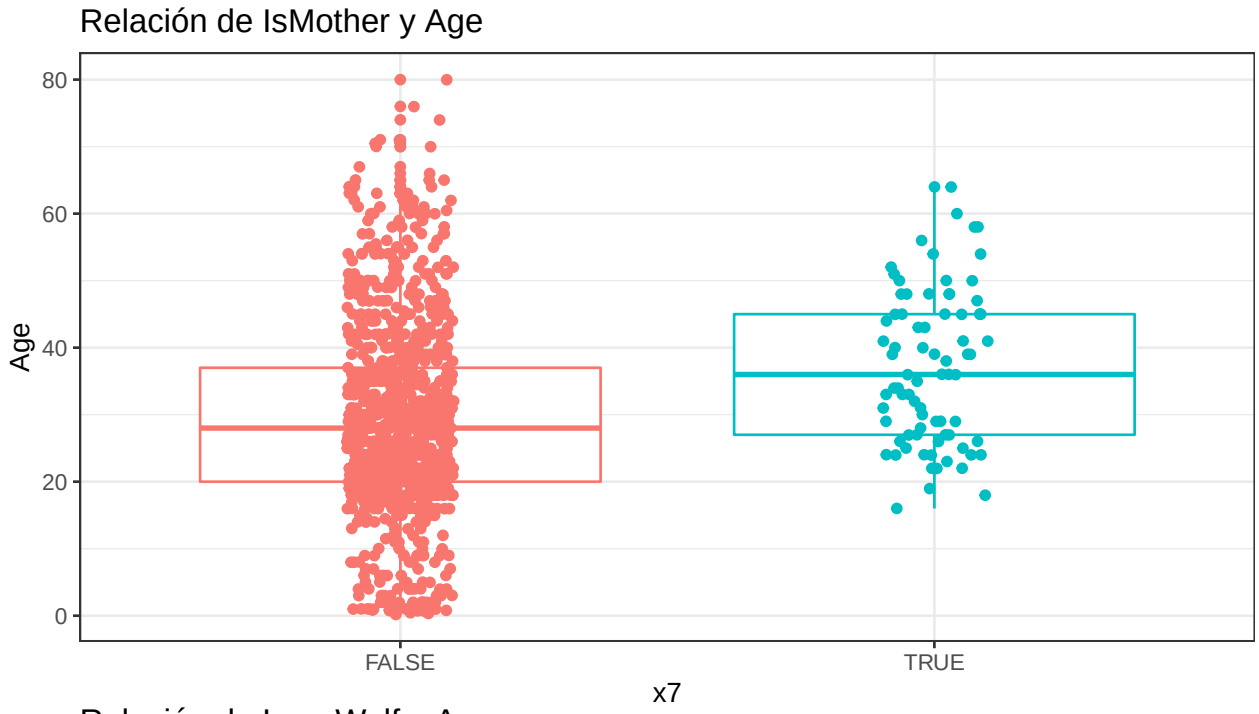


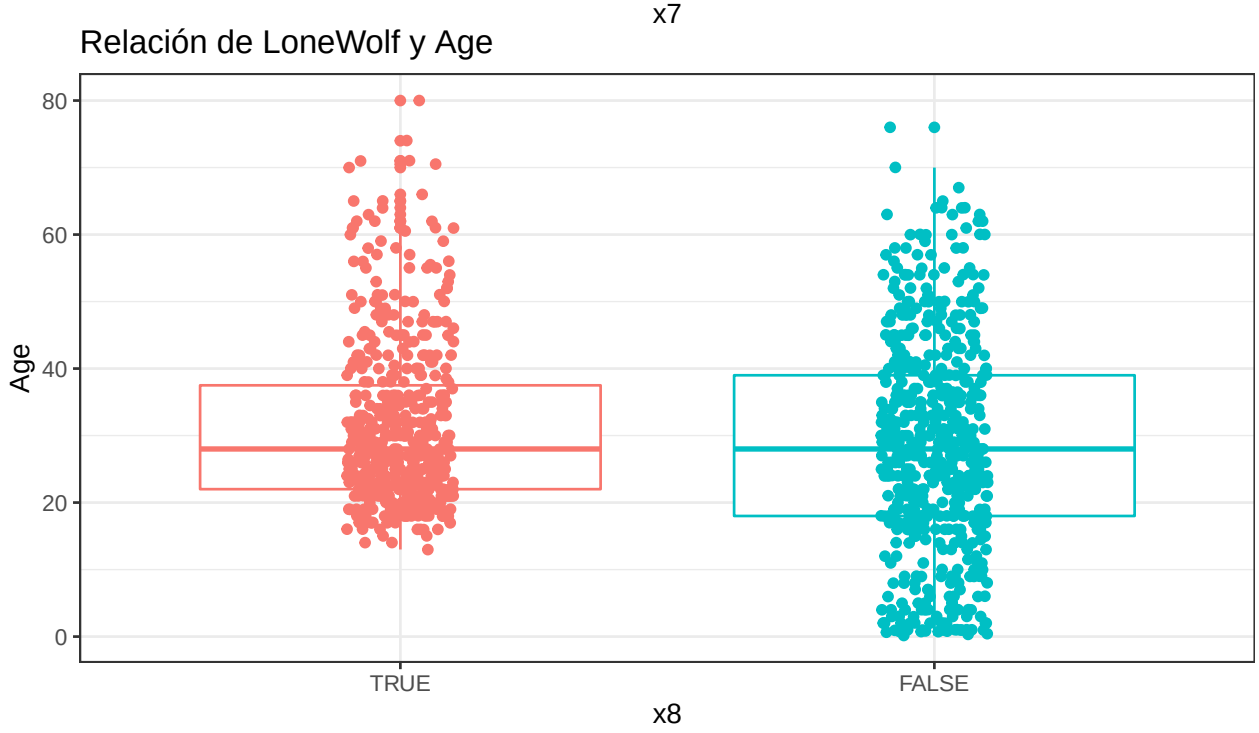

Histograma de edad
Observaciones
Age
0
5
10
15
0
20
40
60

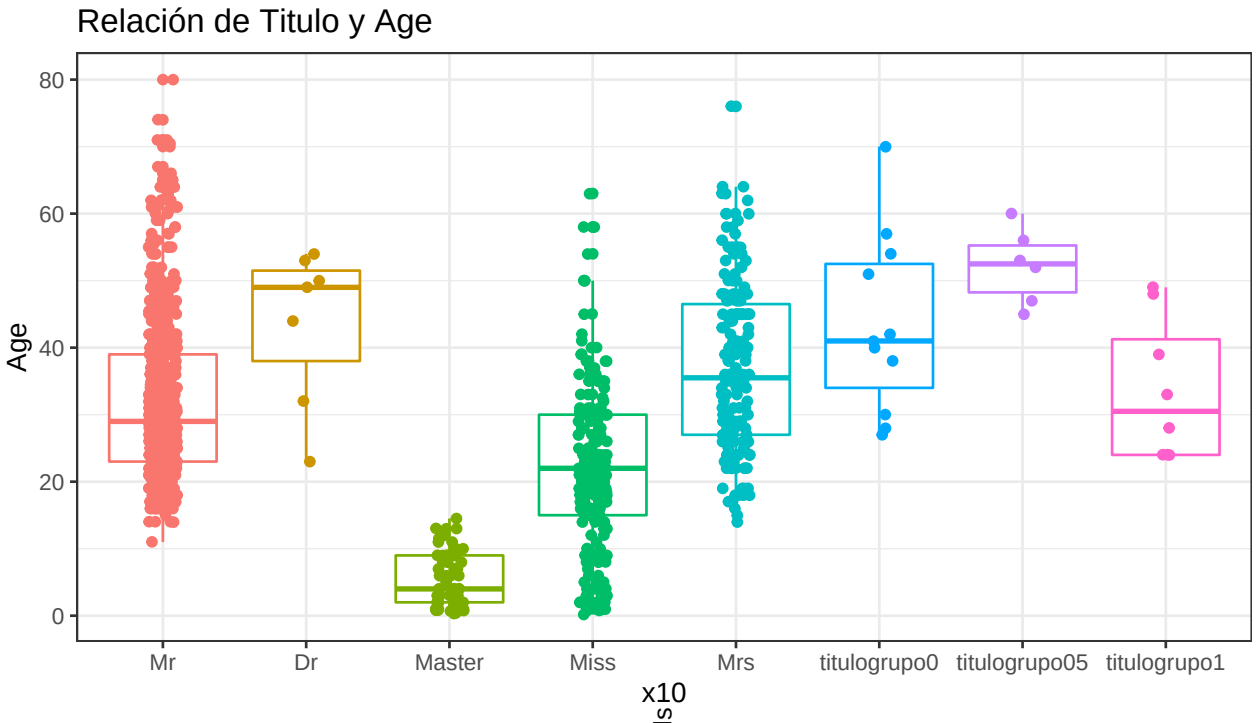

Pclass
SibSp
Parch
Fare
FamSize

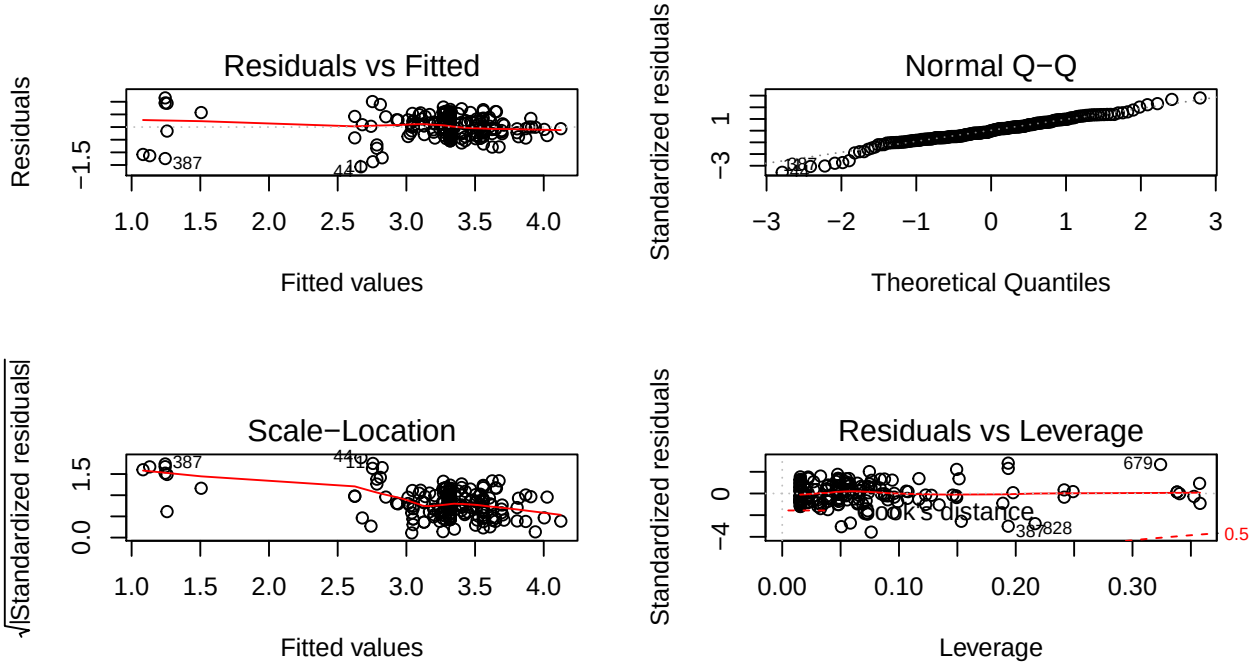

Relación de Sex y Age
Age
80
60
40
20
0
male
female
x5


Relación de Embarked y Age
Age
80
60
40
20
0
S
C
Q
x6


Relación de IsMother y Age
80
60
40
20
0
Age
FALSE
TRUE
x7


Relación de LoneWolf y Age
80
60
40
20
0
Age
TRUE
FALSE
x8


Relación de Titulo y Age
Age
80
60
40
20
0
Mr
Dr
Master
Miss
Mrs
titulogrupo0
titulogrupo05
titulogrupo1


x10
Residuals vs Fitted
Residuals
Fitted values
Normal Q−Q
Standardized residuals
Theoretical Quantiles
Scale−Location
√|Standardized residuals|
Fitted values
Residuals vs Leverage
Standardized residuals
Cook's distance
Leverage

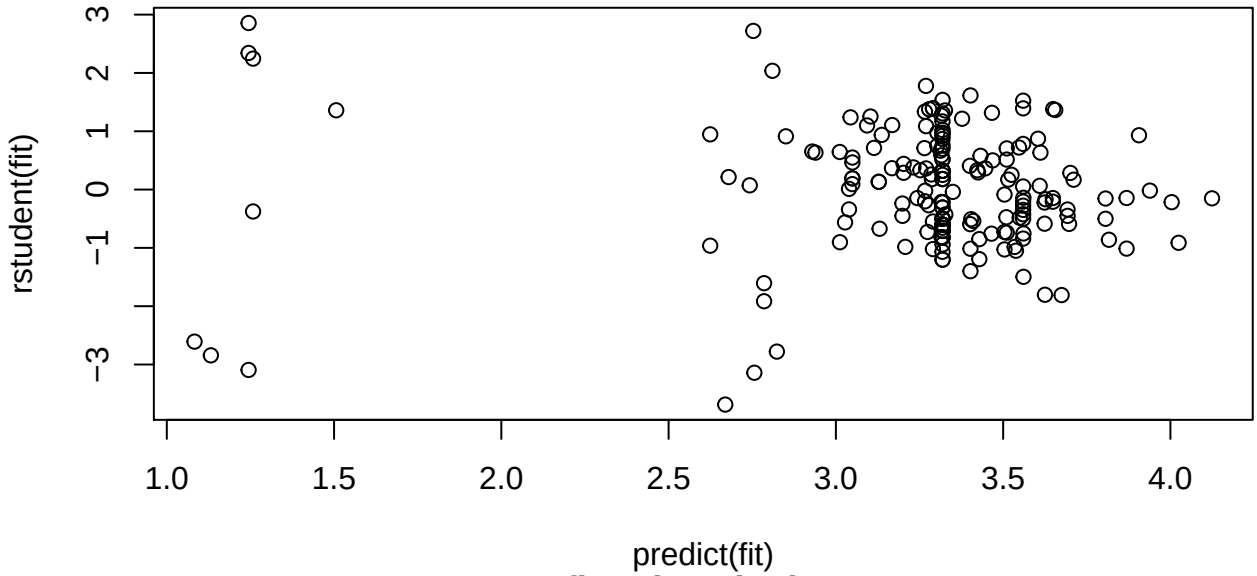
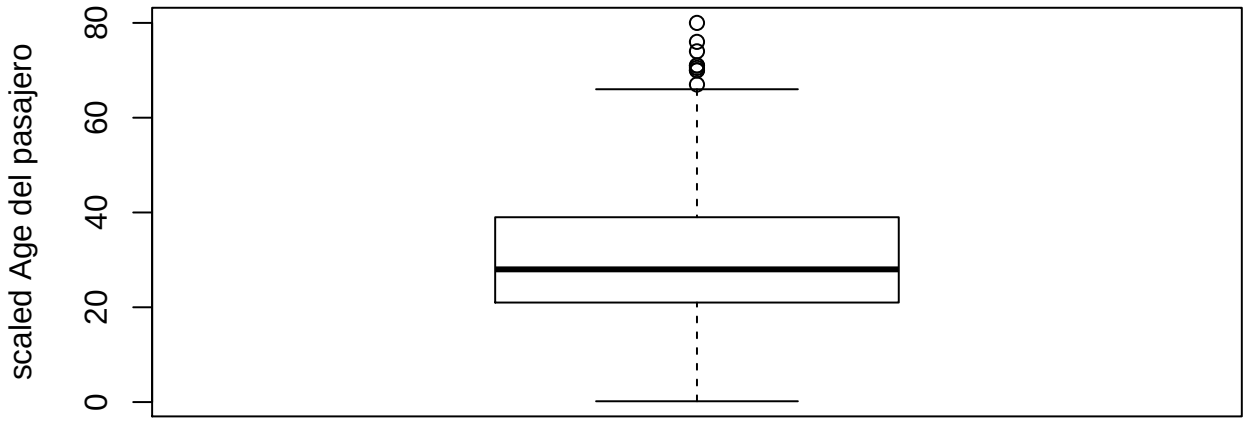
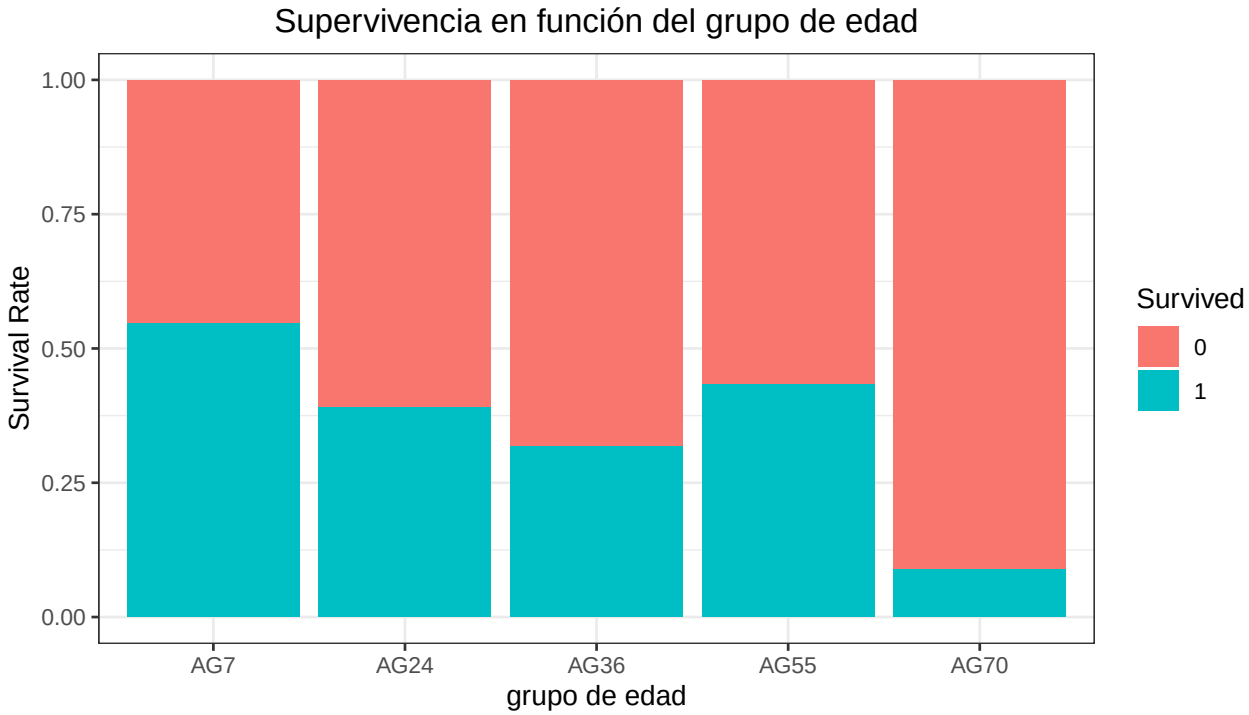
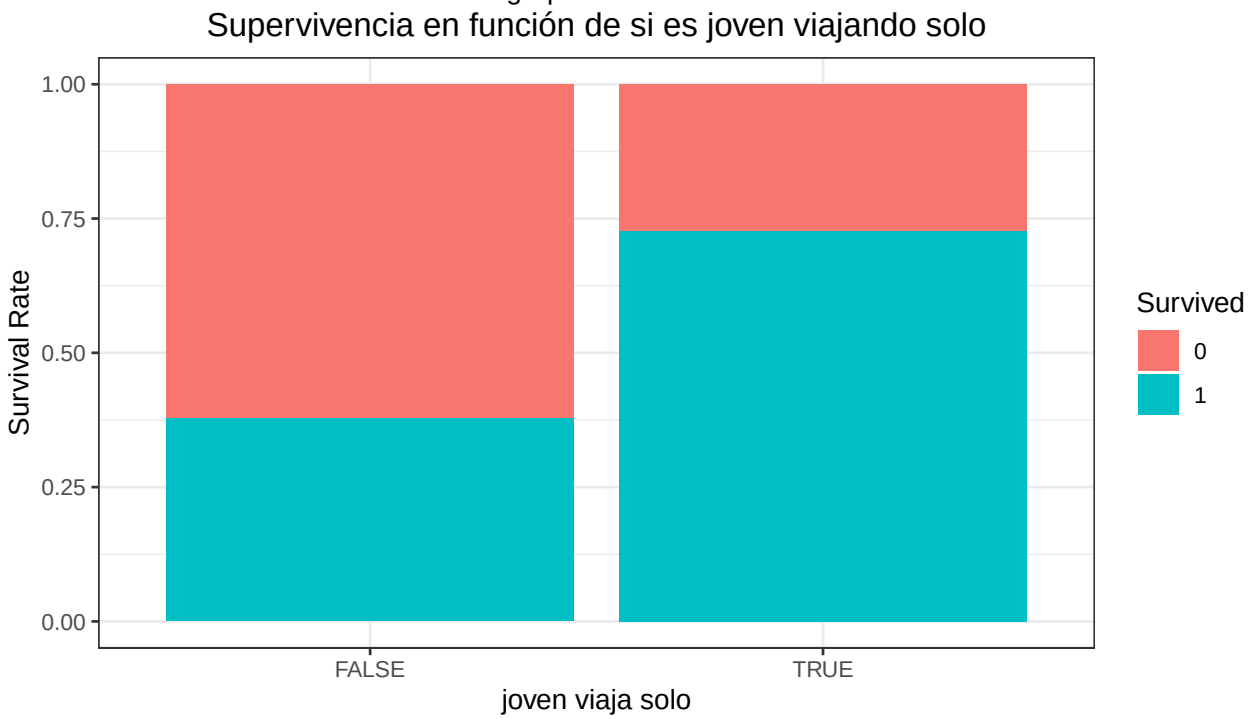


grafico de caja de Age

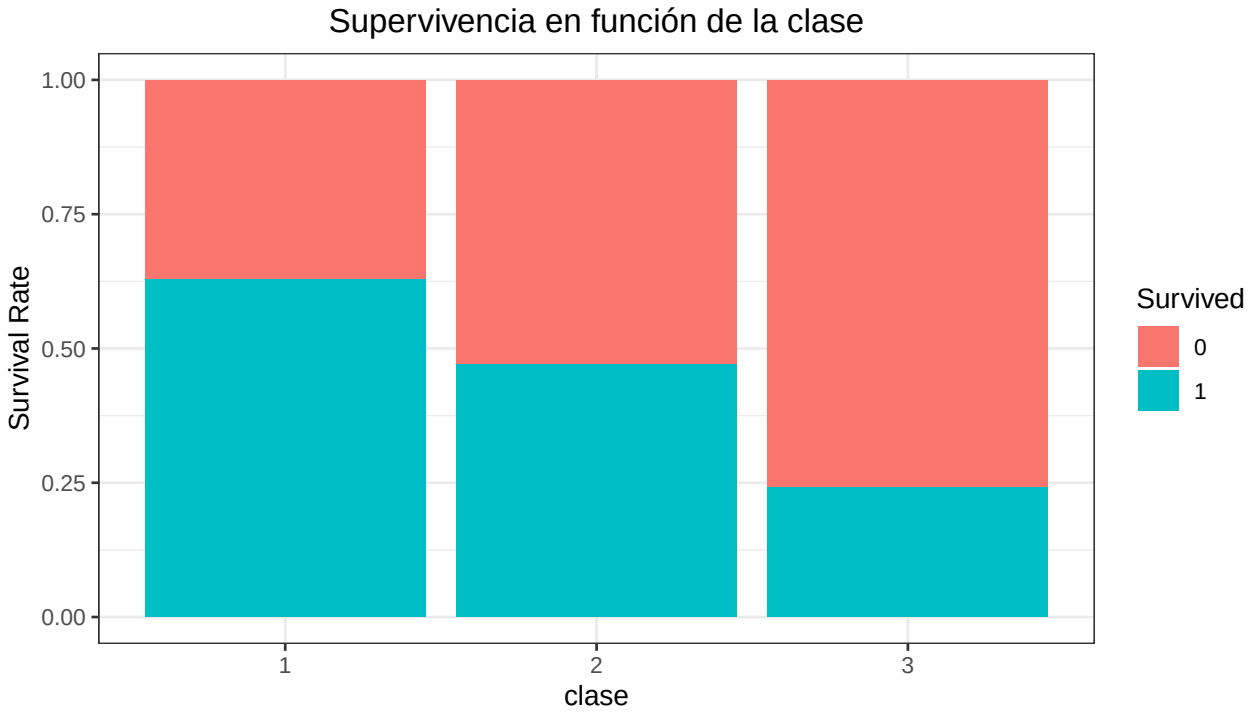






Trabajo con Pclass

Pclas está como integer pero realmente es una categórica, podemos crear una nueva varable tipo factor PclassFactor



3. Limpieza de los datos.

3.1. ¿Los datos contienen ceros o elementos vacíos? ¿Cómo gestionarías cada uno de estos casos?

En la variable Ticket: Hemos tenido que corregir 4 casos de valores perdidos en nuestra nueva variable debido a que estos casos no tenian número de ticket y solo aparecía la palabra LINE. Como solo nos interesa para una variable que agrupa los tamaños de número de ticket le asignamos un 0 en dicha variable, en la variable original no hacemos nada porque no va a ser utilizada.

En la bariable Embarked: Nos encontramos varios valores perdidos que sustituimos por el valor más frecuente.

En la variable Cabina: Existen demasiados valores perdidos y no tenemos una forma buena de predecirlos, sin que produzacan demasiado ruido, por lo que vamos a ignorar esta variable.

En la variable Fare: Existen ciertos pasajeros con Fare 0, pero los vamos a dejar como están.

En la variable Age: Limpiamos los valores perdidos, esto lo vamos a hacer de varias maneras y luego estudiaremos cual es la mejor: - El primer método de imputación será imputar por la edar media del titulo de la persona. Creamos la variable Ages. - El segundo método es KNN con el que imputamos los valores perdidos de Age en función de las variables “Pclass”, “Sex”, “SibSp”, “Parch”, “Fare”, “Embarked”, “Titulo”, ‘FamSize’. Creamos la variable KNNAges - El tercer método es por regresión. Primero vemos como se ditribuye la edad visualizando su histograma con un mayor número de divisiones. Representamos la relación entre las variables cuantitativas y también mostramos la relación entre las variables cualitativas y la edad. Generamos un modelo para predecir el logaritmo de la edad(esto mejora la falta de normalidad de la variable Age) a partir de las variables Pclass, SibSp, Parch, Fare, Sex, Embarked, IsMother, LoneWolfs, FamSize, Titulo y usamos stepAIC para que seleccione el modelo que mejor funciona, partir de esas variables se va quedando con las variables más representativas. Visualizamos los gráficos de los residuos del modelo y creamos la variable RegresionAges para almacenar las edades imputadas.

3.2. Identificación y tratamiento de valores extremos.

En la variable Fare: Para solucionar el problema de los valores extremos vamos agrupamos Fare en categorías y añadimos los valores extremos a la última categoría.

En la variable Age: Para solucionar el problema de los valores extremos agrupamos las edades e incluimos a los valores extremos en la última categoría.

4. Análisis de los datos.

4.1. Selección de los grupos de datos que se quieren analizar/comparar (planificación de los análisis a aplicar).

Inicialmente ponemos las varaiables en grupos en función de la relación que tienen por como se han creado:

“Fare”, “FareGroups”, “FareIndividual”, “FareIndGroups”, “FareIndividualBySex”, “FareIndividualBySexGroups”, ‘FarePw2’, ‘FarePw3’

“Name”,“Titulo”,“FamilyIDTKGrouped”,“FamSize”,“TicketFreq”,“IsMother”

“SibSp”,“Parch”,“FamSize”,“IsMother”,“LoneWolfs”,“FamilyIDTKGrouped”,“youngTravelAl”

“Ticket”,“TicketFreq”,“TktNum”,“tamanoticket”,“FamilyIDTKGrouped”

“Age”,“ageGroupsByCut”,“Ages”,“KNNAges”,“RegresionAges”,“LoneWolfs”,‘AgePw2’,‘AgePw3’,“youngTravelAl”

“Sex”,“FareIndividualBySex”,“FareIndividualBySexGroups”

“Pclass”

“Embarked”

Hemos probado diferentes métodos de análisis para evaluar la eficacia de las variables:

Análisis de las variables por métodos de filtrado

Este análisis se basa en contrastar la correlación de la variables, estudiar por contrastes de hipótesis si la variables independientes tienen alguna variable que es dependiente entre las variables explicativas o viendo la information.gain de las variables(paquete FSelector).

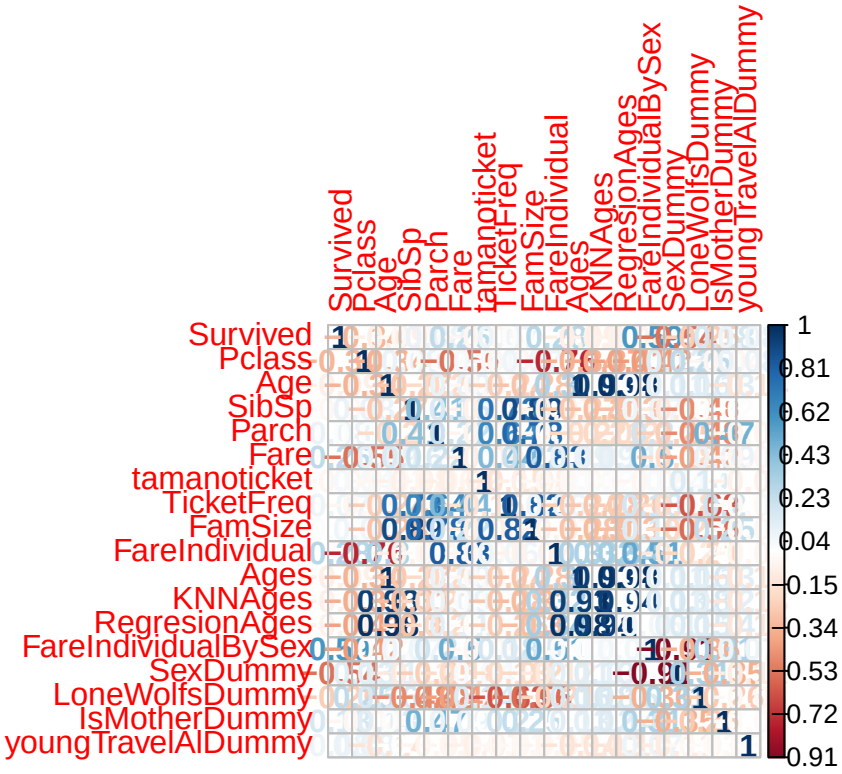
Vamos a analizar la asociacion entre variables de nuestros grupos y entre las variables y Survived:

Mejor relacion con survived por grupos: “FareIndividualBySex”,“FareIndividualBySexGroups”,“FareIndividual” “IsMother”,“TicketFreq” “LoneWolfs”,“IsMother” “TicketFreq” “LoneWolfs”,“ageGroupsByCut”,“Age” “FareIndividualBySex”,“FareIndividualBySexGroups”,“Sex”

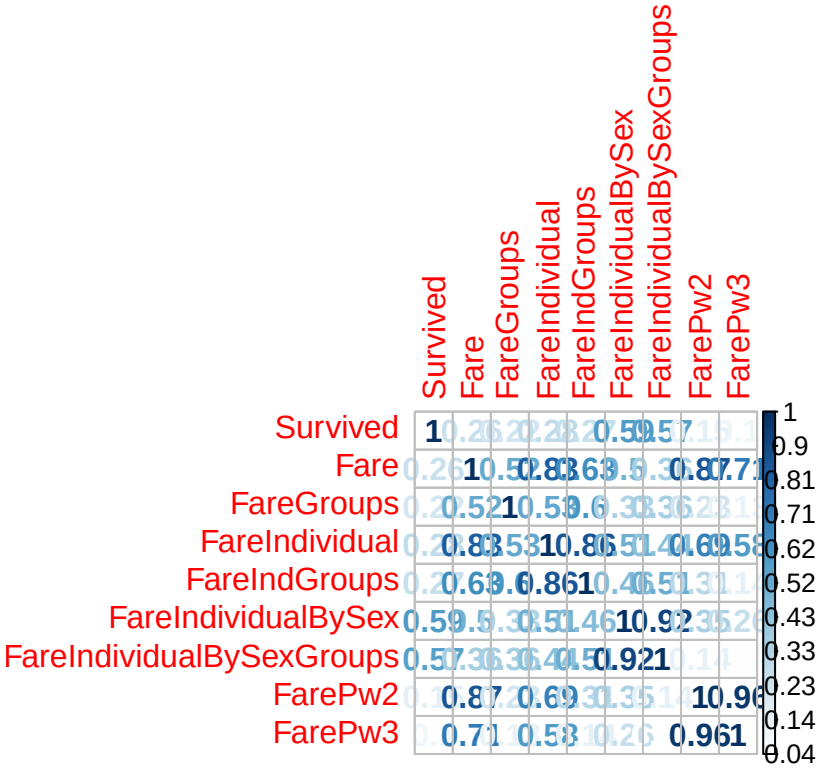
Grupos de variables interesantes tras este análisis: “FareIndividualBySex”,“IsMother”,“LoneWolfs”,“TicketFreq”,“ageGroupsBy “FareIndividualBySexGroups”,“IsMother”,“LoneWolfs”,“TicketFreq”,“ageGroupsByCut”,“Embarked”,“Pclass”,‘FarePw2’ “FamilyIDTKGrouped”,“Titulo”,“FareIndividual”,“IsMother”,“LoneWolfs”,“youngTravelAl”,“Age”,“Embarked”,“Pclass”

Mostrámos gráficamente las relaciones existentes

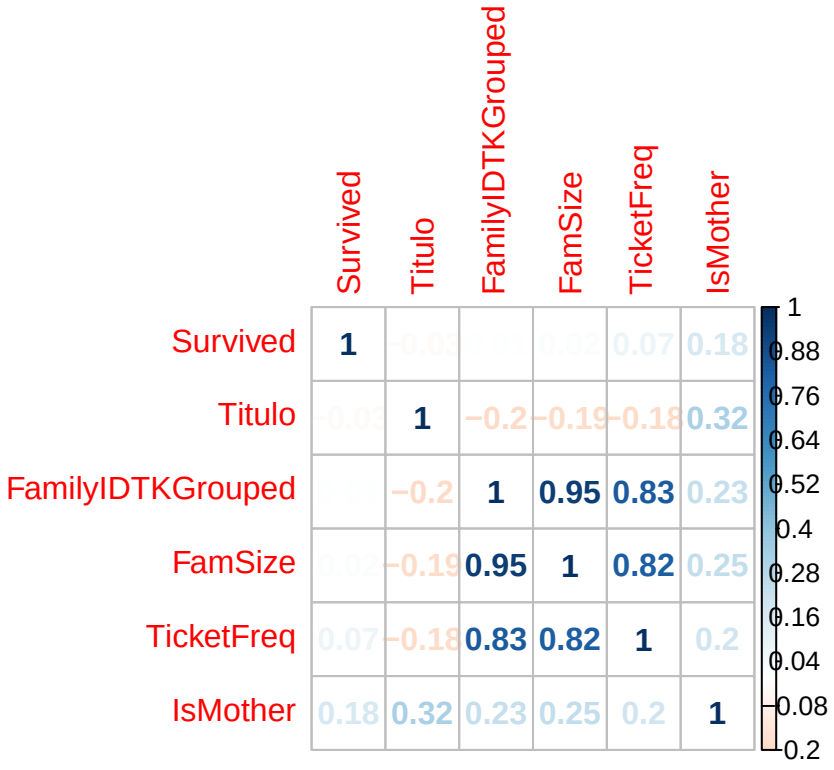
```
## [1] "Mostramos matriz de correlaciones entre las variables numéricas independientes"
```



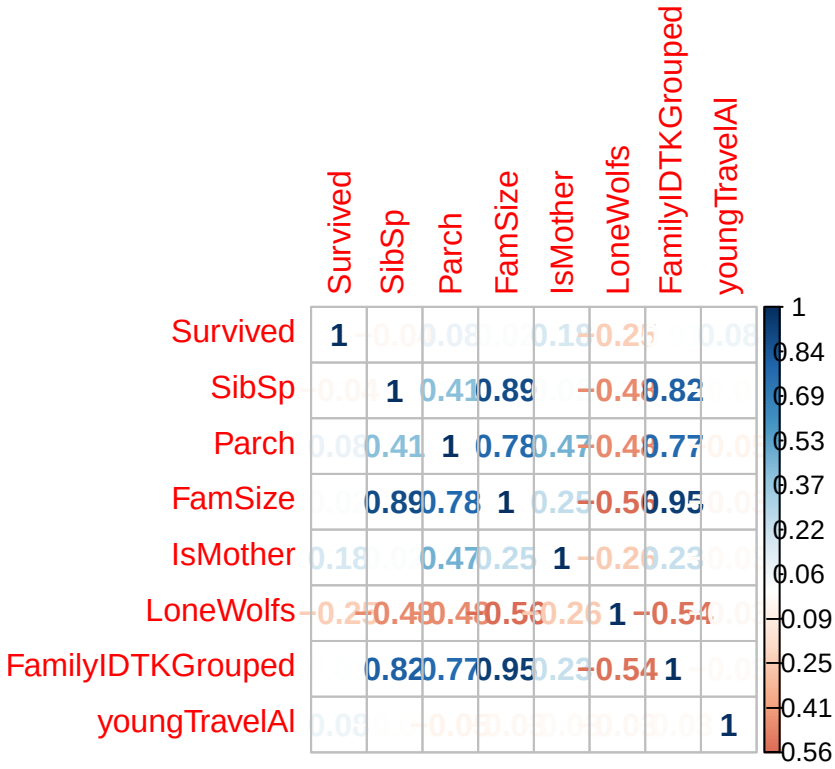

```
## [1] "Mostramos matriz de correlaciones entre las variables del grupo1"
```



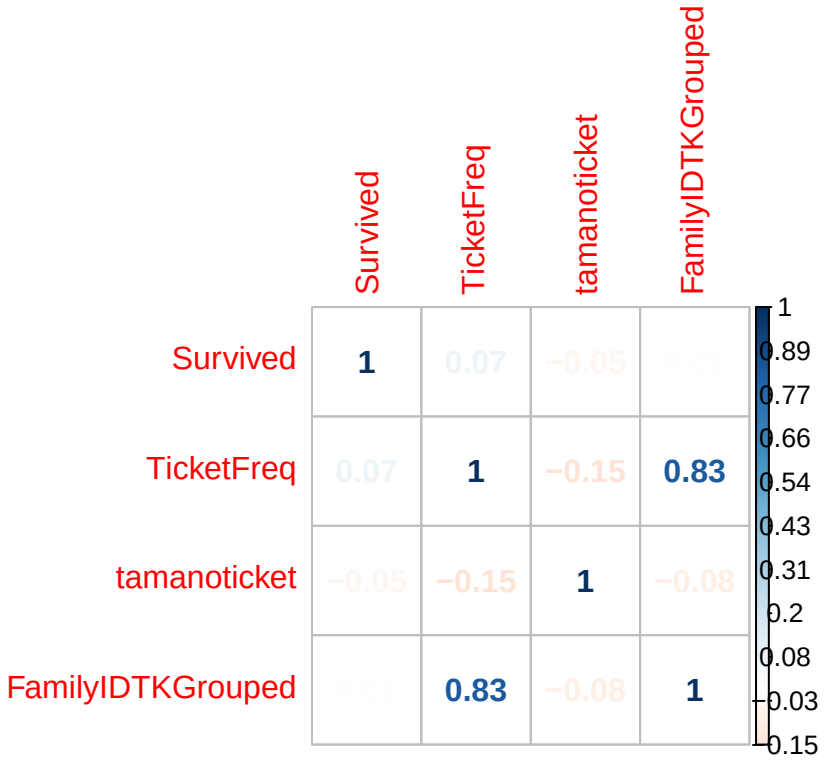

```
## [1] "Mostramos matriz de correlaciones entre las variables del grupo2"
```



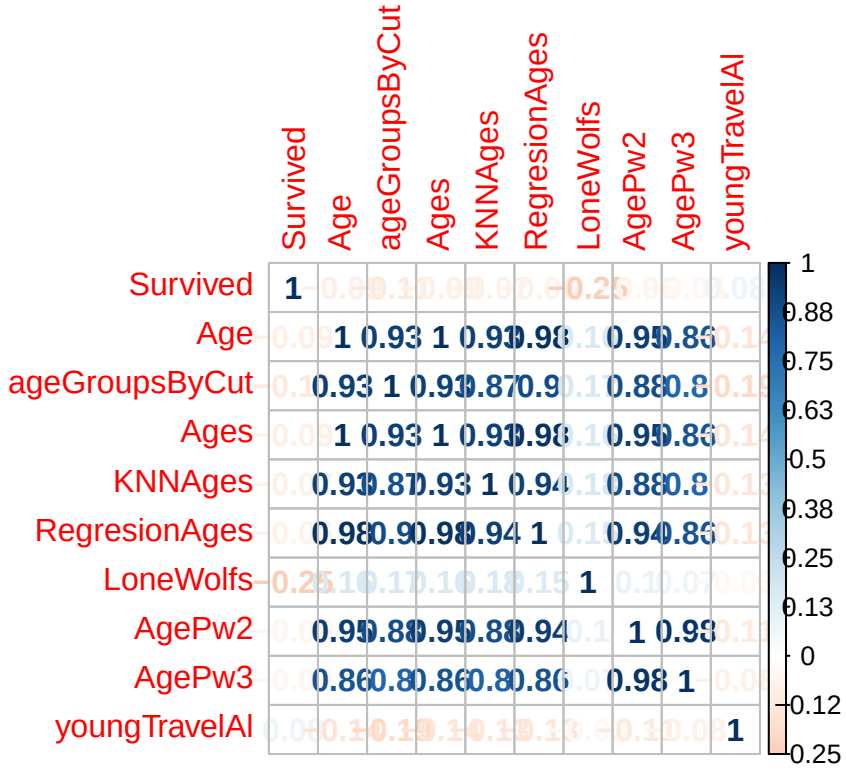

```
## [1] "Mostramos matriz de correlaciones entre las variables del grupo3"
```



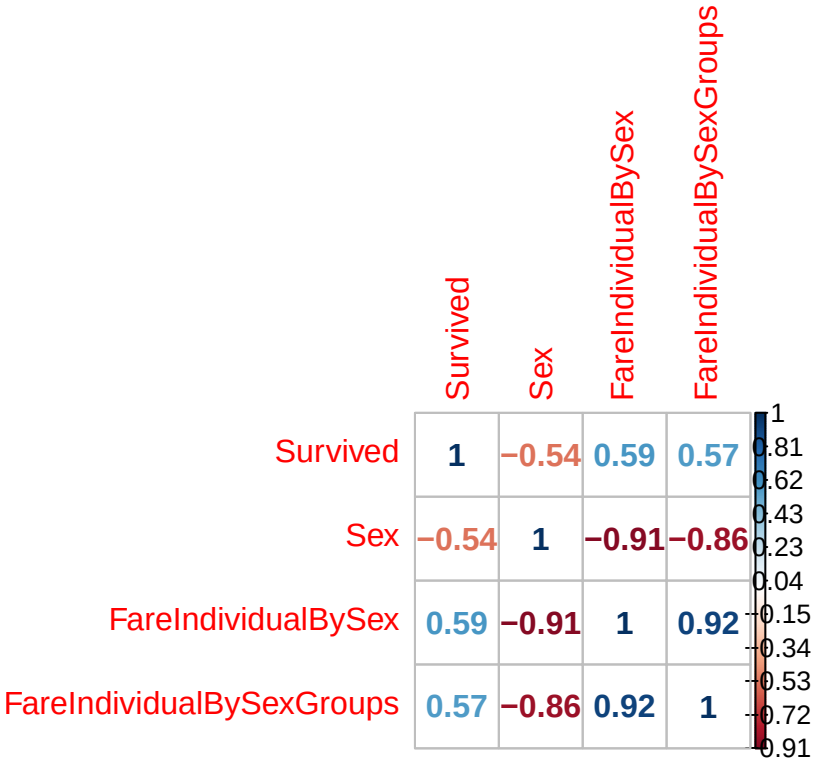

```
## [1] "Mostramos matriz de correlaciones entre las variables del grupo4"
```




```
## [1] "Mostramos matriz de correlaciones entre las variables del grupo5"
```



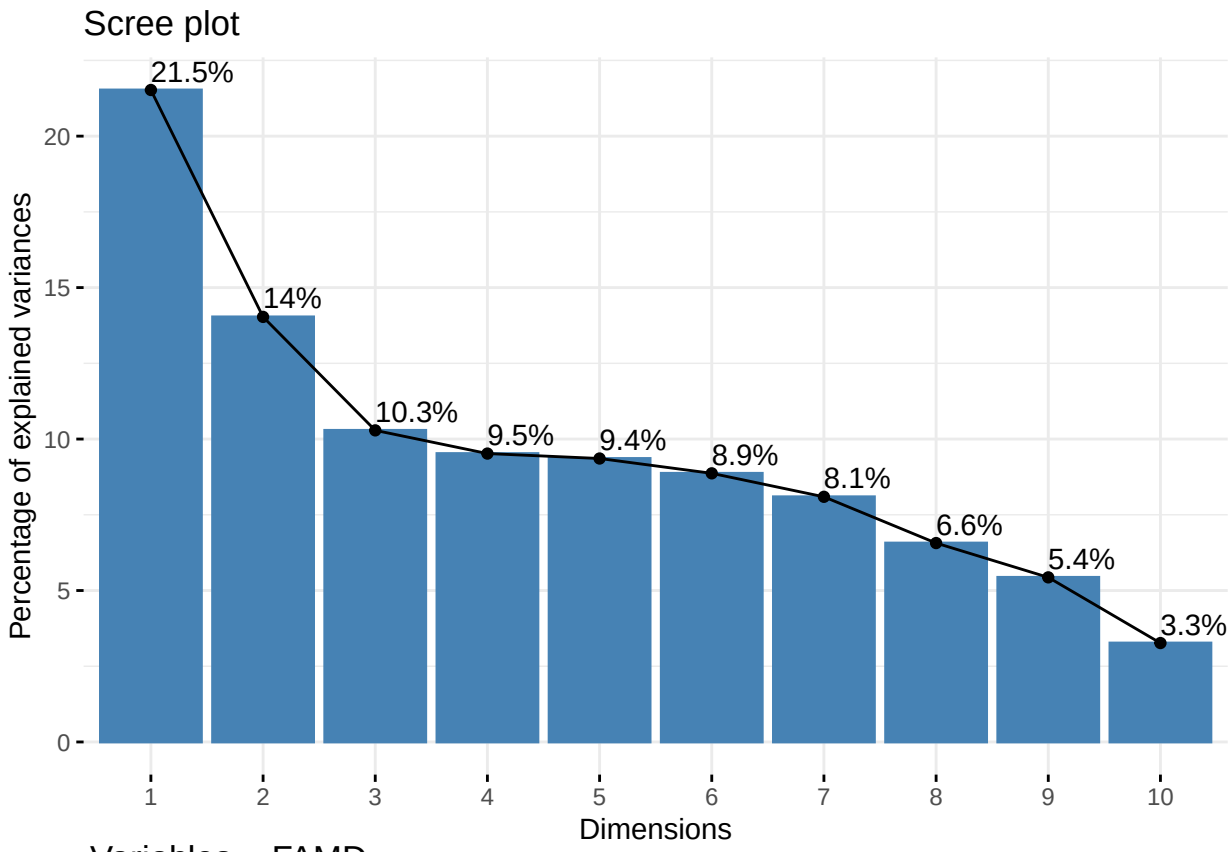

```
## [1] "Mostramos matriz de correlaciones entre las variables del grupo6"
```

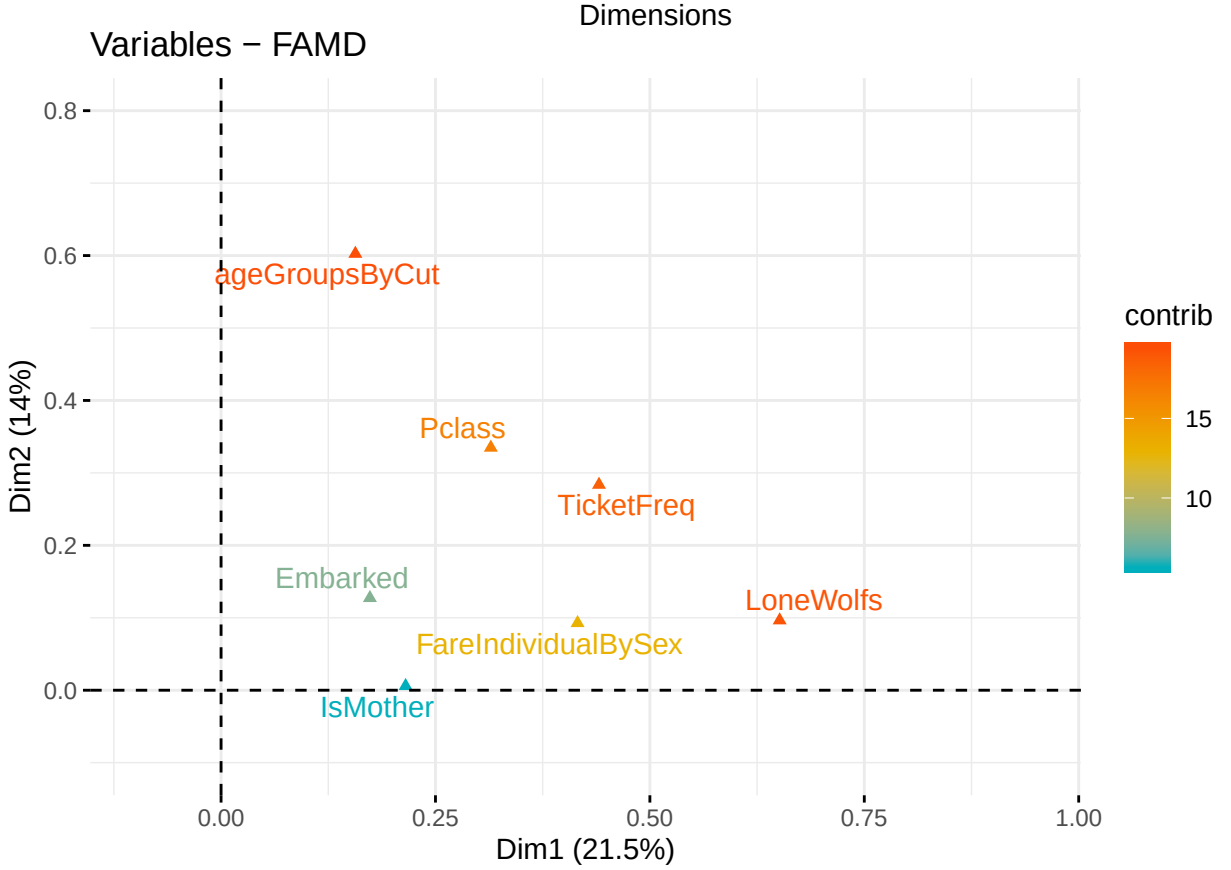


Análisis de las variables por análisis de fatores

El análisis de factores es similar al análisis de componetes principales, le aportamos un conjunto de variables y construye un nuevo conjunto de datos con nuevas variables que representan a los datos originales. Los conjuntos de datos obtenidos con este sistema no nos han dado buenos resultados.

```
##         percentage of variance cumulative percentage of variance
## comp 1                21.522923                          21.52292
## comp 2                14.032248                          35.55517
## comp 3                10.287000                          45.84217
## comp 4                 9.520746                          55.36292
## comp 5                 9.358165                          64.72108
## comp 6                 8.868513                          73.58959
## comp 7                 8.093699                          81.68329
## comp 8                 6.566345                          88.24964
## comp 9                 5.434094                          93.68373
## comp 10                3.265686                          96.94942
## comp 11                3.050581                         100.00000
```


Scree plot
21.5%
14%
10.3%
9.5%
9.4%
8.9%
8.1%
6.6%
5.4%
3.3%
Percentage of explained variances
0
5
10
15
20
1
2
3
4
5
6
7
8
9
10
Dimensions


Variables − FAMD
ageGroupsByCut
Pclass
TicketFreq
Embarked
LoneWolfs
FareIndividualBySex
IsMother
Dim2 (14%)
0.0
0.2
0.4
0.6
0.8
0.00
0.25
0.50
0.75
1.00
Dim1 (21.5%)
contrib
10
15

Análisis de las variables por Forward Selection, Backward Selection, Stepwise Selection

En este método generamos un modelo de regresión logística y mediante stepAIC va optimizando el modelo y eliminando las variables poco significativas.

Los mejores resultados fueron aportados por Titulo + FareIndividual + Pclass + FamilyIDTKGrouped, que arrojó un 85% de acierto en subconjuntos de prueba y test creados a partir del conjunto de entrenamiento para tener una forma de estimar el porcentaje de acierto(posteriormente descubrí que este porcentaje de acierto no es buen indicador).

Análisis de las variables por Backward Selection con paquete mlr

Este método nos selecciona las variables más importantes del dataset, vamos a utilizar Backward selection que parte de todas las variables y va eliminando las menos significativas.

Variables recomendadas por este método: “Pclass”, “FareIndividualBySex”, “ageGroupsByCut” “Pclass”, “Titulo”, “FamSize”, “FareIndGroups”, “FareIndividualBySex”, “AgePw2” “Pclass”, “KNNAgesScaled”, “FareIndividualBySexScaled” “SibSp”, “FareIndividualBySex”, “AgePw3”, “PclassScaled”

```
## Features         : 6
## Performance      : mmce.test.mean=0.1784512
## Pclass, TicketFreq, FareIndividualBySex, AgePw2, FarePw3, Master
##
## Path to optimum:
## - Features:   78  Init   :                       Perf = 0.17845  Diff: NA  *
## - Features:   77  Remove : FamSize               Perf = 0.17845  Diff: 0  *
## - Features:   76  Remove : titulogrupo05         Perf = 0.17845  Diff: 0  *
## - Features:   75  Remove : Q                     Perf = 0.17845  Diff: 0  *
## - Features:   74  Remove : Parch                 Perf = 0.17845  Diff: 0  *
## - Features:   73  Remove : FIBSG323              Perf = 0.17845  Diff: 0  *
## - Features:   72  Remove : Miss                  Perf = 0.17845  Diff: 0  *
## - Features:   71  Remove : AG55                  Perf = 0.17845  Diff: 0  *
## - Features:   70  Remove : FIG247                Perf = 0.17845  Diff: 0  *
## - Features:   69  Remove : RegresionAges         Perf = 0.17845  Diff: 0  *
## - Features:   68  Remove : FareIndGroups         Perf = 0.17845  Diff: 0  *
## - Features:   67  Remove : Ages                  Perf = 0.17845  Diff: 0  *
## - Features:   66  Remove : AG70                  Perf = 0.17845  Diff: 0  *
## - Features:   65  Remove : FIG512                Perf = 0.17845  Diff: 0  *
## - Features:   64  Remove : KNNAges               Perf = 0.17845  Diff: 0  *
## - Features:   63  Remove : AG7                   Perf = 0.17845  Diff: 0  *
## - Features:   62  Remove : titulogrupo0          Perf = 0.17845  Diff: 0  *
## - Features:   61  Remove : Sex                   Perf = 0.17845  Diff: 0  *
## - Features:   60  Remove : titulogrupo1          Perf = 0.17845  Diff: 0  *
## - Features:   59  Remove : FIBSG20               Perf = 0.17845  Diff: 0  *
## - Features:   58  Remove : FIBSG64               Perf = 0.17845  Diff: 0  *
## - Features:   57  Remove : LoneWolfs             Perf = 0.17845  Diff: 0  *
## - Features:   56  Remove : FareIndividual        Perf = 0.17845  Diff: 0  *
## - Features:   55  Remove : Size4                 Perf = 0.17845  Diff: 0  *
## - Features:   54  Remove : Size2                 Perf = 0.17845  Diff: 0  *
## - Features:   53  Remove : FIBSG148              Perf = 0.17845  Diff: 0  *
## - Features:   52  Remove : FIG135                Perf = 0.17845  Diff: 0  *
## - Features:   51  Remove : C                     Perf = 0.17845  Diff: 0  *
## - Features:   50  Remove : FG235                 Perf = 0.17845  Diff: 0  *
## - Features:   49  Remove : FG262                 Perf = 0.17845  Diff: 0  *
## - Features:   48  Remove : Size3                 Perf = 0.17845  Diff: 0  *
## - Features:   47  Remove : FIG56                 Perf = 0.17845  Diff: 0  *
## - Features:   46  Remove : FarePw2               Perf = 0.17845  Diff: 0  *
```

```
## - Features:   45  Remove : FIG15                Perf = 0.17845  Diff: 0  *
## - Features:   44  Remove : FIBSG247             Perf = 0.17845  Diff: 0  *
## - Features:   43  Remove : AgePw3               Perf = 0.17845  Diff: 0  *
## - Features:   42  Remove : Fare                 Perf = 0.17845  Diff: 0  *
## - Features:   41  Remove : FG512                Perf = 0.17845  Diff: 0  *
## - Features:   40  Remove : PclassFactor         Perf = 0.17845  Diff: 0  *
## - Features:   39  Remove : FareGroups           Perf = 0.17845  Diff: 0  *
## - Features:   38  Remove : Mr                   Perf = 0.17845  Diff: 0  *
## - Features:   37  Remove : FIG141               Perf = 0.17845  Diff: 0  *
## - Features:   36  Remove : Age                  Perf = 0.17845  Diff: 0  *
## - Features:   35  Remove : FIBSG51              Perf = 0.17845  Diff: 0  *
## - Features:   34  Remove : FG13                 Perf = 0.17845  Diff: 0  *
## - Features:   33  Remove : AG36                 Perf = 0.17845  Diff: 0  *
## - Features:   32  Remove : FIBSG14              Perf = 0.17845  Diff: 0  *
## - Features:   31  Remove : FG47                 Perf = 0.17845  Diff: 0  *
## - Features:   30  Remove : tamanoticket         Perf = 0.17845  Diff: 0  *
## - Features:   29  Remove : Size5                Perf = 0.17845  Diff: 0  *
## - Features:   28  Remove : SibSp                Perf = 0.17845  Diff: 0  *
## - Features:   27  Remove : Embarked             Perf = 0.17845  Diff: 0  *
## - Features:   26  Remove : youngTravelAl        Perf = 0.17845  Diff: 0  *
## - Features:   25  Remove : Titulo               Perf = 0.17845  Diff: 0  *
## - Features:   24  Remove : FIG121               Perf = 0.17845  Diff: 0  *
## - Features:   23  Remove : FG112                Perf = 0.17845  Diff: 0  *
## - Features:   22  Remove : Mrs                  Perf = 0.17845  Diff: 0  *
## - Features:   21  Remove : IsMother             Perf = 0.17845  Diff: 0  *
## - Features:   20  Remove : Size7                Perf = 0.17845  Diff: 0  *
## - Features:   19  Remove : Dr                   Perf = 0.17845  Diff: 0  *
## - Features:   18  Remove : Size6                Perf = 0.17845  Diff: 0  *
## - Features:   17  Remove : FG78                 Perf = 0.17845  Diff: 0  *
## - Features:   16  Remove : FareIndividualByS... Perf = 0.17845  Diff: 0  *
## - Features:   15  Remove : AG24                 Perf = 0.17845  Diff: 0  *
## - Features:   14  Remove : S                    Perf = 0.17845  Diff: 0  *
## - Features:   13  Remove : SizeBig              Perf = 0.17845  Diff: 0  *
## - Features:   12  Remove : ageGroupsByCut       Perf = 0.17845  Diff: 0  *
## - Features:   11  Remove : Alone                Perf = 0.17845  Diff: 0  *
## - Features:   10  Remove : PassengerId          Perf = 0.17845  Diff: 0  *
## - Features:    9  Remove : FG164                Perf = 0.17845  Diff: 0  *
## - Features:    8  Remove : FG214                Perf = 0.17845  Diff: 0  *
## - Features:    7  Remove : FIBSG512             Perf = 0.17845  Diff: 0  *
## - Features:    6  Remove : FG142                Perf = 0.17845  Diff: 0  *
##
## Stopped, because no improving feature was found.
```

Análisis de las variables por Recursive Feature Elimination Method (RFE)

Este método es similar al anterior pero también obtenemos un listado con las variables ordenadas por importancia

Variables recomendadas por este método: “FareIndividualBySex”, “Titulo”, “FareIndividual”, “KNNAges”, “Sex” FareIndividualBySex, Titulo, KNNAgesScaled

A partir de la lista de importancia de las variables, obtenemos las más importantes que no estén relacionadas entre si: “FareIndividualBySex”,“Titulo”,“KNNAges”,‘PclassScaled’,‘FamilyIDTKGrouped’,‘Embarked’,‘LoneWolfsDummy’,‘youn

```
##                              Overall
## FareIndividualBySex       13.600949
## FareIndividualBySexScaled 13.582484
## Titulo                    11.363564
```

```
## SexDummy                    8.709066
## KNNAgesScaled               8.703271
## KNNAges                     8.636648
## FareIndividualScaled        8.454495
## FareIndividual              8.335605
## Sex                         8.206359
## AgePw3                      8.007264
## Ages                        7.957815
## PclassScaled                7.956500
## AgePw2                      7.909015
## FarePw2                     7.904552
## FareScaled                  7.886852
## Age                         7.867687
## Pclass                      7.855544
## Fare                        7.839743
## AgesScaled                  7.827489
## AgeScaled                   7.809420
## FarePw3                     7.754604
## PclassFactor                7.513338
## FareIndividualBySexGroups   7.468243
## RegresionAgesScaled         7.398841
## RegresionAges               7.226660
## FamilyIDTKGrouped           6.101545
## TicketFreq                  5.329217
## TicketFreqScaled            5.297692
## Embarked                    4.907116
## FamSize                     4.225986
## FamSizeScaled               4.124099
## ageGroupsByCut              3.835092
## FareIndGroups               3.317044
## SibSp                       3.287760
## SibSpScaled                 3.198566
## FareGroups                  3.127860
## LoneWolfsDummy              2.815112
## LoneWolfs                   2.677530
## PassengerId                 1.720031
## youngTravelAl               1.499671
## IsMother                    1.492384
## IsMotherDummy               1.479338
## tamanoticket                1.476044
## youngTravelAlDummy          1.449961
## tamanoticketScaled          1.343119
## ParchScaled                 1.131183
## Parch                       1.066165
```

Análisis de las variables RandomForest

En este método utilizamos random forest, que genera árboles de decisión aleatorios y va comparando los resultados obtenidos para filtrar las variables más importantes.

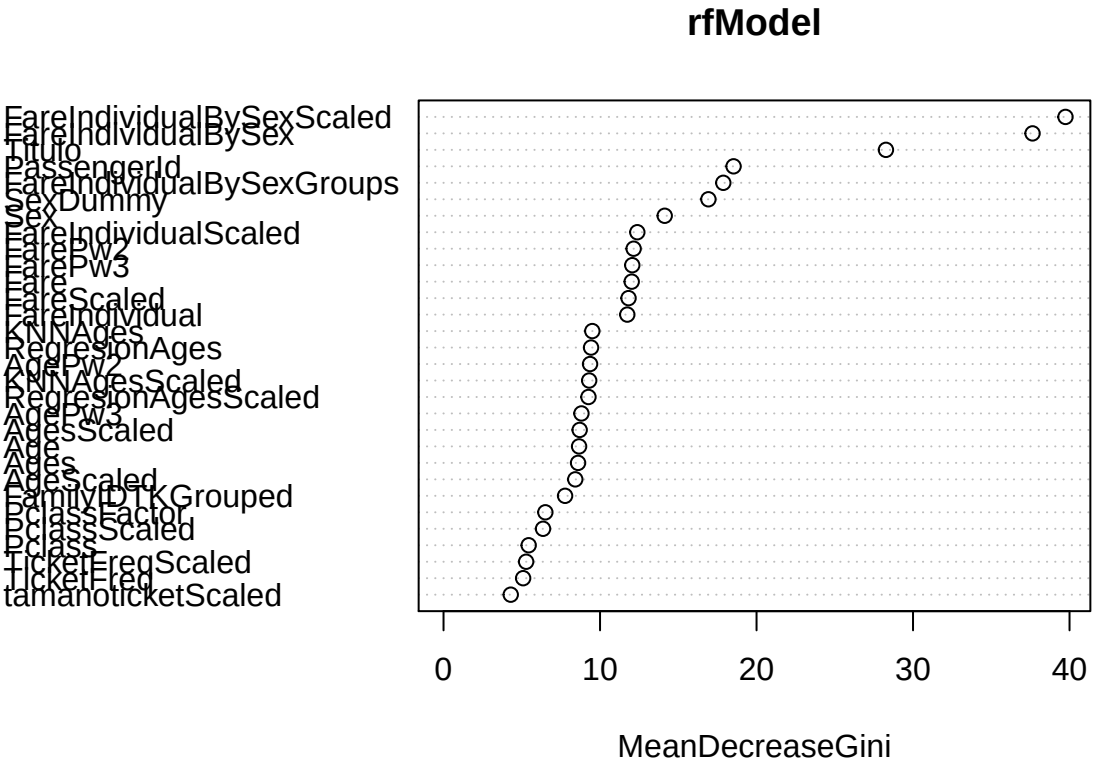
Variables recomendadas por este método: 'FareIndividualBySex','Titulo','Mr','KNNAges','FIBSG14','FamilyIDTKGrouped','Pc
'FareIndividualBySex','Titulo','KNNAges','FamilyIDTKGrouped','PclassFactor','tamanoticket','Embarked','LoneWolfsDummy'
'FareIndividualBySex','Titulo','ageGroupsByCut','FamilyIDTKGrouped','PclassFactor','tamanoticket','Embarked','LoneWolfsD

Esta última selección de componentes es la que mejores resultados nos ha dado de todos los grupos de variables obtenidos en todos los análisis.

```
## [1] "Importancia de las variables"
```

```
##                              MeanDecreaseGini
## PassengerId                        18.5322089
## Pclass                              5.4413971
## Sex                                14.1321532
## Age                                 8.6606190
## SibSp                               2.8868562
## Parch                               1.1570235
## Fare                               12.0225108
## Embarked                            2.8644493
## tamanoticket                        4.1010606
## TicketFreq                          5.0860676
## Titulo                             28.2684083
## FamSize                             3.8478243
## LoneWolfs                           0.7540797
## FamilyIDTKGrouped                   7.7660895
## IsMother                            0.4410108
## FareGroups                          2.5226443
## FareIndividual                     11.7421928
## FareIndGroups                       3.5352225
## FareIndividualBySex                37.6356491
## FareIndividualBySexGroups          17.8752323
## ageGroupsByCut                      2.0976972
## Ages                                8.5962136
## KNNAges                             9.5059961
## RegresionAges                       9.4265706
## youngTravelAl                       0.2369829
## PclassFactor                        6.5061408
## AgePw2                              9.3639177
## FarePw2                            12.1469618
## AgePw3                              8.8112974
## FarePw3                            12.0595040
## PclassScaled                        6.3584410
## AgeScaled                           8.4181092
## SibSpScaled                         2.6542732
## ParchScaled                         1.2044630
## FareScaled                         11.8217099
## tamanoticketScaled                  4.2943865
## TicketFreqScaled                    5.2798304
## FamSizeScaled                       4.1272630
## FareIndividualScaled               12.3820225
## AgesScaled                          8.6994244
## KNNAgesScaled                       9.3075847
## RegresionAgesScaled                 9.2619201
## FareIndividualBySexScaled          39.7404963
## SexDummy                           16.9248790
## LoneWolfsDummy                      1.2194964
## IsMotherDummy                       0.4884667
## youngTravelAlDummy                  0.3449758

## [1] "Importancia de las variables 2"
```



4.2. Comprobación de la normalidad y homogeneidad de la varianza.

La normalidad la asumimos por el teorema central del limite, ya que la muestra es muy superior a 30 elementos, lo que nos permite asumir que las variables son normales.

4.3. Aplicación de pruebas estadísticas para comparar los grupos de datos. En función de los datos y el objetivo del estudio, aplicar pruebas de contraste de hipótesis, correlaciones, regresiones, etc. Aplicar al menos tres métodos de análisis diferentes.

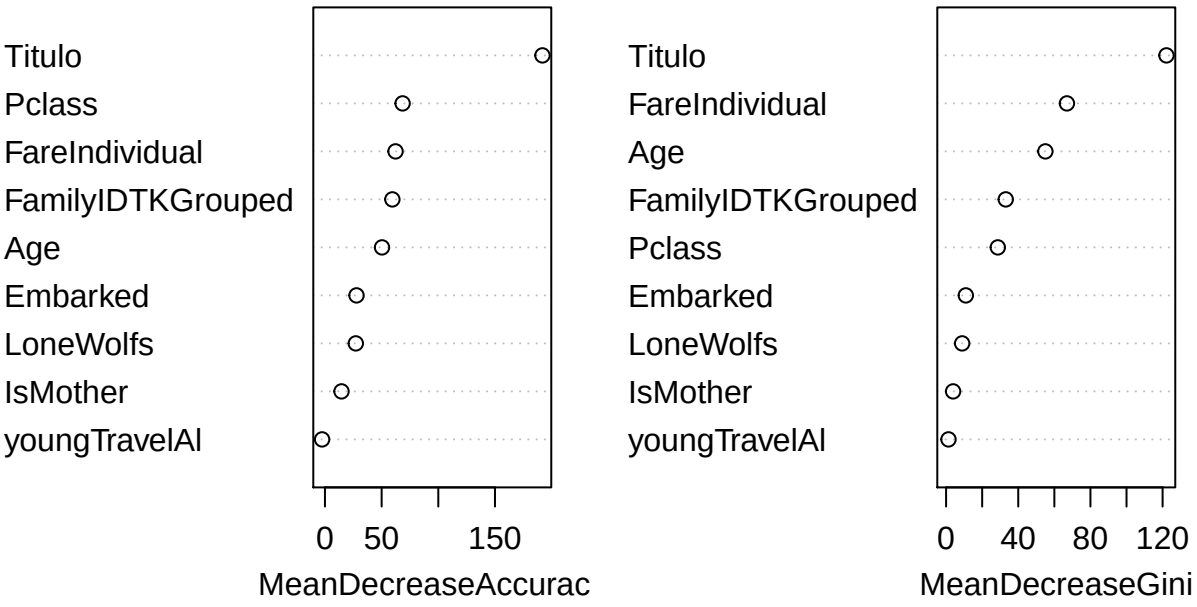
Vamos a utilizar diferentes métodos para predecir los resultados

Prediccion por Random Forest

Este método lo aplicamos para predecir a partir de las variables:

'PassengerId','Survived',"FamilyIDTKGrouped","Titulo","FareIndividual","IsMother","LoneWolfs","youngTravelAl","Age","E
- el resultado obtenido lo subimos a kaggle y nos dió: 0.779

'FareIndividualBySex','Titulo','ageGroupsByCut','FamilyIDTKGrouped','PclassFactor','tamanoticket','Embarked','LoneWolfsD
- en kaggle obtuvimos un 0.794



Predicción por KNN

Este método obtiene los valores de Survived en funcion de sus vecinos más proximos que encuentra a partir de la distáncia euclídea. Para usar este método escalamos todas las variables cuantitativas y pasamos todas las variables cualitativas a dummy, creando una variable numérica por cada uno de los factores de las variables.

El resultado obtenido por este método fue subido a kaggle y dió: 0.732

Predicción por regresión logística

Hemos probado diferentes modelos con las sugerencias de los análisis sin demasiado éxito. Por este método hemos obtenido en kaggle un 0.784 a partir de las siguientes variables: FareIndividualBySexScaled + IsMotherDummy + LoneWolfsDummy + TicketFreqScaled + ageGroupsByCut + Embarked+PclassScaled

```
##
## Call:
## glm(formula = Survived ~ FareIndividualBySexScaled + IsMotherDummy +
##     LoneWolfsDummy + TicketFreqScaled + ageGroupsByCut + Embarked +
##     PclassScaled, family = binomial, data = train2)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4079  -0.5897  -0.3893   0.6484   2.8909
##
## Coefficients:
##                            Estimate Std. Error z value Pr(>|z|)
## (Intercept)                  0.9606     0.3000   3.203 0.001362 **
## FareIndividualBySexScaled    1.3200     0.1110  11.892  < 2e-16 ***
## IsMotherDummy                0.6951     0.3888   1.788 0.073757 .
```

```
## LoneWolfsDummy             -0.2635     0.2438  -1.081 0.279773
## TicketFreqScaled           -0.4594     0.1391  -3.303 0.000955 ***
## ageGroupsByCutAG24         -1.5306     0.3289  -4.653 3.26e-06 ***
## ageGroupsByCutAG36         -1.9423     0.3511  -5.532 3.17e-08 ***
## ageGroupsByCutAG55         -2.3656     0.4647  -5.091 3.57e-07 ***
## ageGroupsByCutAG70         -3.7876     1.1512  -3.290 0.001002 **
## EmbarkedC                   0.2802     0.2351   1.192 0.233346
## EmbarkedQ                   0.4537     0.3260   1.392 0.164009
## PclassScaled               -0.6049     0.1070  -5.654 1.57e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1186.7  on 890  degrees of freedom
## Residual deviance:  787.9  on 879  degrees of freedom
## AIC: 811.9
##
## Number of Fisher Scoring iterations: 5

## [1] "Porcentaje de corte óptimo"

## [1] 0.5869116

## [1] "Error por fallo de mala clasificación corte= 0.58691162565729"

## [1] 0.193

## [1] "Error por fallo de mala clasificación corte 0.5"

## [1] 0.2009

## [1] "Tabla de contingencia de predicciones contra observaciones Survived"

##         as.factor(predicciones)
## Survived No Sobrevive Sobrevive
##        0          465        84
##        1           95       247

##         0         1
## 0.8469945 0.7222222

## [1] "Matriz de porcentage de aciertos del modelo"

## [1] "acierto general del modelo"

## [1] 0.7991021
```

Table 2: Matriz de acierto del modelo. acierto global: 0.799102132435466

	(Prediccion peso)		
	No Sobrevive	Sobrevive	porcentaje_de_acierto
0	465	84	0.8469945
1	95	247	0.7222222

```
## [1] "--------------------------------------------------------------------------------"

## [1] "Tabla de contingencia de predicciones optimizadas peso contra observaciones peso"

##         as.factor(prediccionesOptimizadas)
## Survived No Sobrevive Sobrevive
##        0          502        47
##        1          125       217

## [1] "--------------------------------------------------------------------------------"
```

```
## [1] "--------------------------------------------------------------------------------"
```

```
## [1] "Porcentaje de acierto general del modelo:  0.80695847362514"
```

```
## [1] "Matriz de porcentage de aciertos del modelo optimizado"
```

Table 3: Matriz de acierto del modelo optimizado. acierto global: 0.80695847362514

	(Prediccion peso)		
	No Sobrevive	Sobrevive	porcentaje_de_acierto
0	502	47	0.9143898
1	125	217	0.6345029

```
## [1] "variance inflation factors"
```

```
## FareIndividualBySexScaled             IsMotherDummy            LoneWolfsDummy
##                    1.2141                    1.3121                    1.8684
##          TicketFreqScaled       ageGroupsByCutAG24       ageGroupsByCutAG36
##                    1.9932                    3.3705                    3.4301
##       ageGroupsByCutAG55       ageGroupsByCutAG70                EmbarkedC
##                    2.1091                    1.1140                    1.0717
##                EmbarkedQ             PclassScaled
##                    1.1315                    1.4733
```

```
## [1] "loglikelihod modelo"
```

```
## 'log Lik.' -393.9477 (df=12)
```

```
## Likelihood ratio test
##
## Model 1: Survived ~ FareIndividualBySexScaled + IsMotherDummy + LoneWolfsDummy +
##     TicketFreqScaled + ageGroupsByCut + Embarked + PclassScaled
## Model 2: Survived ~ 1
##   #Df  LogLik  Df  Chisq Pr(>Chisq)
## 1  12 -393.95
## 2   1 -593.33 -11 398.76  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
## [1] "loglikelihod Ratio test contra modelo con solo la constante"
```

```
## Likelihood ratio test
##
## Model 1: Survived ~ 1
## Model 2: Survived ~ FareIndividualBySexScaled + IsMotherDummy + LoneWolfsDummy +
##     TicketFreqScaled + ageGroupsByCut + Embarked + PclassScaled
##   #Df  LogLik Df  Chisq Pr(>Chisq)
## 1   1 -593.33
## 2  12 -393.95 11 398.76  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
## [1] "MacFadden's Pseudo R²"
```

```
## 'log Lik.' 0.3360367 (df=12)
```

```
## Wald test
##
## Model 1: Survived ~ 1
## Model 2: Survived ~ FareIndividualBySexScaled + IsMotherDummy + LoneWolfsDummy +
##     TicketFreqScaled + ageGroupsByCut + Embarked + PclassScaled
##   Res.Df Df  Chisq Pr(>Chisq)
## 1    890
```

```
## 2    879 11 238.86  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
## Call:
## concordance.formula(object = train2$Survived ~ predicciones)
##
## n= 891
## Concordance= 0.7846 se= 0.01434
## concordant discordant     tied.x     tied.y    tied.xy
##     114855       7980      64923      62525     146212
```

```
## Overall accuracy = 0.799
##
## Confusion matrix
##       Predicted (cv)
## Actual     0     1
##      0 0.847 0.153
##      1 0.278 0.722
```

```
## [1] "Matriz de confusion a partir de los datos originales"
```

```
## $overall
## [1] 0.7991021
##
## $confusion
##       Predicted (cv)
## Actual         0         1
##      0 0.8469945 0.1530055
##      1 0.2777778 0.7222222
##
## $prior
##         0         1
## 0.6161616 0.3838384
```

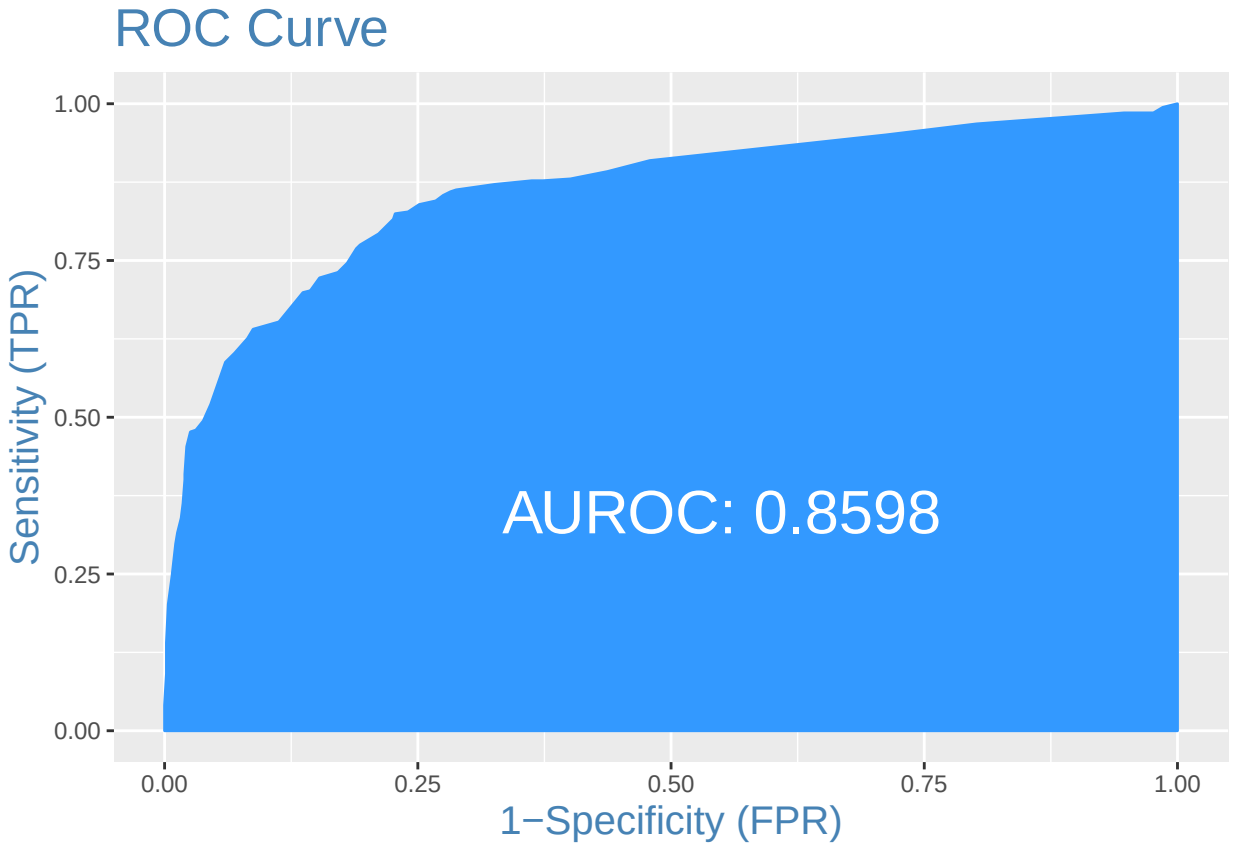
```
## [1] "Odds ratio calculadas a partir de los coeficientes"
```

```
##               (Intercept) FareIndividualBySexScaled             IsMotherDummy
##                2.61338795               3.74336540               2.00399717
##            LoneWolfsDummy          TicketFreqScaled        ageGroupsByCutAG24
##                0.76834242               0.63166906               0.21639608
##        ageGroupsByCutAG36        ageGroupsByCutAG55        ageGroupsByCutAG70
##                0.14337267               0.09388892               0.02265068
##                 EmbarkedC                EmbarkedQ              PclassScaled
##                1.32332882               1.57412683               0.54614746
```

```
##
##  Hosmer and Lemeshow goodness of fit (GOF) test
##
## data:  mymod3$y, fitted(mymod3)
## X-squared = 15.684, df = 8, p-value = 0.04713
```

```
##                  y0 y1    yhat0     yhat1
## [0.00691,0.0705] 97  9 99.89115  6.108845
## (0.0705,0.101]   66  8 67.43804  6.561959
## (0.101,0.105]    79  9 78.97161  9.028389
## (0.105,0.151]    79 12 79.98772 11.012283
## (0.151,0.268]    77 10 69.46005 17.539949
## (0.268,0.453]    52 37 55.97799 33.022015
## (0.453,0.58]     51 38 41.97969 47.020307
## (0.58,0.717]     34 55 31.39176 57.608243
## (0.717,0.88]     10 79 17.44657 71.553430
```

```
## (0.88,0.997]      4 85  6.45542 82.544580
```



Prediccion por redes neuronales y análisis de componetes principales

Aquí usamos una red neuronal que incluye un análisis de componentes principales(pcaNNet). Hemos probado a realizar los cálculos con todo el dataset para ver si el análisis de componentes principales obtenía un buen resultado, pero no fue así. Usando todo el dataset solo obtuvimos un 0.72 en Kaggle.

Después optamos por usar alguno de los conjuntos de variables selecionados en el análisis y obtuvimos mejores resultados, llegando a 0.784.

```
# # Partimos el dataset en los conjuntos iniciales de train y test.
# trainCARET <- muestra[1:891,-c(4,9,11,13,14,20,54)]
# testCARET <- muestra[892:1309,-c(4,9,11,13,14,20,54)]
#
# # #he function createDataPartition can be used to create a stratified random sample of the data into training and
# # #library(caret)
# # set.seed(998)
# # inTraining <- createDataPartition(trainCARET$Survived, p = .75, list = FALSE)
# # trainingCARET <- trainCARET[ inTraining,]
# # testingCARET  <- trainCARET[-inTraining,]
#
# trainingMod<-trainCARET
# trainingMod$Survived<-as.factor(trainingMod$Survived)
# levels(trainingMod$Survived) <- c('No','Si')
# levels(trainingMod$PclassFactor) <- c('Primera','Segunda','Tercera')
# trainingMod$LoneWolfs<-as.logical(trainingMod$LoneWolfs)
# trainingMod$LoneWolfs <- ifelse(trainingMod$LoneWolfs, 'Si', 'No')
# trainingMod$IsMother<-as.logical(trainingMod$IsMother)
```

```
# trainingMod$IsMother <- ifelse(trainingMod$IsMother, 'Si', 'No')
# trainingMod$youngTravelAl<-as.logical(trainingMod$youngTravelAl)
# trainingMod$youngTravelAl  <- ifelse(trainingMod$youngTravelAl, 'Si', 'No')
# trainingMod$LoneWolfs <- as.factor(trainingMod$LoneWolfs)
# trainingMod$IsMother <- as.factor(trainingMod$IsMother)
# trainingMod$youngTravelAl  <- as.factor(trainingMod$youngTravelAl)
# trainingMod$Survived  <- as.factor(trainingMod$Survived)
#
# testMod<-testCARET
# testMod$LoneWolfs<-as.logical(testMod$LoneWolfs)
# testMod$LoneWolfs <- ifelse(testMod$LoneWolfs, 'Si', 'No')
# testMod$IsMother<-as.logical(testMod$IsMother)
# testMod$IsMother <- ifelse(testMod$IsMother, 'Si', 'No')
# testMod$youngTravelAl<-as.logical(testMod$youngTravelAl)
# testMod$youngTravelAl  <- ifelse(testMod$youngTravelAl, 'Si', 'No')
# testMod$LoneWolfs <- as.factor(testMod$LoneWolfs)
# testMod$IsMother <- as.factor(testMod$IsMother)
# testMod$youngTravelAl  <- as.factor(testMod$youngTravelAl)
# testMod$Survived  <- as.factor(testMod$Survived)
#
# #creamos un conjunto de entrenamiento y test propios a partir de train para calcular nuestros porcentaje de acier
# ## 85% of the sample size
# smp_size <- floor(0.85 * nrow(trainingMod))
# ## Establecemos una semilla para poder volver a reproducir el mismo muestreo
# set.seed(123)
# train_ind <- sample(seq_len(nrow(trainingMod)), size = smp_size)
# training <- trainingMod[train_ind, ]
# testing <- trainingMod[-train_ind, ]
# test3Survival<-testing$Survived
# testing$Survived <- NA
#
#  train_control <- trainControl(method = 'cv', number = 10, classProbs = TRUE, summaryFunction = twoClassSummary)
# #
# # # #Basic Linear Modelling
# #
# # logRegModel <- caret::train(Survived ~ ., data = training, trControl = train_control,
# #                       method = 'glm', family = binomial(),
# #                       metric = 'ROC')
# # # cross validation results
# # print(logRegModel)
# # coef(summary(logRegModel))
# #
# #
# # #Naive Bayes
# # nbModel <- caret::train(Survived ~ ., data = training, trControl = train_control,
# #                   method = 'nb', tuneLength = 10,
# #                   metric = 'ROC')
# #
# # #Random Forest - Variable Importance
# # rfModel <- caret::train(Survived ~ ., data = training, trControl = train_control,
# #                   method = 'rf', tuneLength = 10,
# #                   metric = 'ROC')
# # varImp(rfModel)
# #
# #Neural Networks
# nnGrid <- expand.grid(.size = c(1,2,3,4,5,6,7),
#                       .decay = c(0, .01, .1, .2, .3, .4, .5, 1, 2))
#
```

```
# nnModel <- caret::train(Survived ~ FareIndividualBySex+IsMother+LoneWolfs+TicketFreq+ageGroupsByCut+Embarked+Pclo
#                  method = 'nnet', tuneGrid = nnGrid,
#                  metric = 'ROC', trace = FALSE)
#
# pcaNNModel <- caret::train(Survived ~ ., data = training, trControl = train_control,
#                  method = 'pcaNNet', tuneGrid = nnGrid,
#                  metric = 'ROC', trace = FALSE)
#
# #models <- list(nn = nnModel, pcann = pcaNNModel)
# # models_preds <- lapply(models, predict, newdata = testing, type = 'prob')
# # models_probs <- as.data.frame(sapply(models_preds, function(df){1 - df$N}))
# # models_probs$Survived <- training$Survived
# nnModelTest <- data.frame(PassengerId = testing$PassengerId,
#                               Survived = ifelse(predict(nnModel, testing) == 'Si',1,0))
# pcaNNModelTest <- data.frame(PassengerId = testing$PassengerId,
#                               Survived = ifelse(predict(pcaNNModel, testing) == 'Si',1,0))
#
#
# #Vamos a comparar el resultado con las observaciones de la variable y construir una matriz de acierto
# t_cont_prediccion_contra_observación<- xtabs(~test3Survival+nnModelTest$Survived)
# print("Tabla de contingencia de predicciones survival contra observaciones survival")
# t_cont_prediccion_contra_observación
# sumatorios_filas <- rowSums(t_cont_prediccion_contra_observación)
# sumatorios_columnas <- colSums(t_cont_prediccion_contra_observación)
# total_elementos <- sum(t_cont_prediccion_contra_observación)
# # obtenemos los porcentages de aciertos de 1s(Sensitivity) y 0s(specificity)
# porcentaje_de_acierto <- t_cont_prediccion_contra_observación[,1]/sumatorios_filas
# porcentaje_de_acierto[2]<-1-porcentaje_de_acierto[2]
# porcentaje_de_acierto
# print("Matriz de porcentage de aciertos del modelo")
# matriz_aciertos<-cbind(t_cont_prediccion_contra_observación,porcentaje_de_acierto)
# print("acierto general del modelo")
# acierto_tortal <-  (t_cont_prediccion_contra_observación[1,1]+t_cont_prediccion_contra_observación[2,2])/total_el
# acierto_tortal
# kable(matriz_aciertos, caption = paste("Matriz de acierto del modelo. acierto global:",acierto_tortal))%>%
# kable_styling(latex_options = c("striped", "hold_position"), position = "center", font_size = 8)%>%
# add_header_above(c(" ", "(Prediccion survival)" = 2," "), bold = T, italic = T)%>%
# pack_rows("(Observacioens survival)", 1, 2)
#
# #Vamos a comparar el resultado con las observaciones de la variable y construir una matriz de acierto
# t_cont_prediccion_contra_observación<- xtabs(~test3Survival+pcaNNModelTest$Survived)
# print("Tabla de contingencia de predicciones survival contra observaciones survival")
# t_cont_prediccion_contra_observación
# sumatorios_filas <- rowSums(t_cont_prediccion_contra_observación)
# sumatorios_columnas <- colSums(t_cont_prediccion_contra_observación)
# total_elementos <- sum(t_cont_prediccion_contra_observación)
# # obtenemos los porcentages de aciertos de 1s(Sensitivity) y 0s(specificity)
# porcentaje_de_acierto <- t_cont_prediccion_contra_observación[,1]/sumatorios_filas
# porcentaje_de_acierto[2]<-1-porcentaje_de_acierto[2]
# porcentaje_de_acierto
# print("Matriz de porcentage de aciertos del modelo")
# matriz_aciertos<-cbind(t_cont_prediccion_contra_observación,porcentaje_de_acierto)
# print("acierto general del modelo")
# acierto_tortal <-  (t_cont_prediccion_contra_observación[1,1]+t_cont_prediccion_contra_observación[2,2])/total_el
# acierto_tortal
# kable(matriz_aciertos, caption = paste("Matriz de acierto del modelo. acierto global:",acierto_tortal))%>%
# kable_styling(latex_options = c("striped", "hold_position"), position = "center", font_size = 8)%>%
# add_header_above(c(" ", "(Prediccion survival)" = 2," "), bold = T, italic = T)%>%
```

```r
# pack_rows("(Observacioens survival)", 1, 2)


## para envio
# Partimos el dataset en los conjuntos iniciales de train y test.
trainCARET <- muestra[1:891,-c(4,9,11,13,14,20,54)]
testCARET <- muestra[892:1309,-c(4,9,11,13,14,20,54)]

trainingMod<-trainCARET
trainingMod$Survived<-as.factor(trainingMod$Survived)
levels(trainingMod$Survived) <- c('No','Si')
levels(trainingMod$PclassFactor) <- c('Primera','Segunda','Tercera')
trainingMod$LoneWolfs<-as.logical(trainingMod$LoneWolfs)
trainingMod$LoneWolfs <- ifelse(trainingMod$LoneWolfs, 'Si', 'No')
trainingMod$IsMother<-as.logical(trainingMod$IsMother)
trainingMod$IsMother <- ifelse(trainingMod$IsMother, 'Si', 'No')
trainingMod$youngTravelAl<-as.logical(trainingMod$youngTravelAl)
trainingMod$youngTravelAl  <- ifelse(trainingMod$youngTravelAl, 'Si', 'No')
trainingMod$LoneWolfs <- as.factor(trainingMod$LoneWolfs)
trainingMod$IsMother <- as.factor(trainingMod$IsMother)
trainingMod$youngTravelAl  <- as.factor(trainingMod$youngTravelAl)
trainingMod$Survived  <- as.factor(trainingMod$Survived)

testMod<-testCARET
testMod$LoneWolfs<-as.logical(testMod$LoneWolfs)
testMod$LoneWolfs <- ifelse(testMod$LoneWolfs, 'Si', 'No')
testMod$IsMother<-as.logical(testMod$IsMother)
testMod$IsMother <- ifelse(testMod$IsMother, 'Si', 'No')
testMod$youngTravelAl<-as.logical(testMod$youngTravelAl)
testMod$youngTravelAl  <- ifelse(testMod$youngTravelAl, 'Si', 'No')
testMod$LoneWolfs <- as.factor(testMod$LoneWolfs)
testMod$IsMother <- as.factor(testMod$IsMother)
testMod$youngTravelAl  <- as.factor(testMod$youngTravelAl)
testMod$Survived  <- NULL
levels(testMod$PclassFactor) <- c('Primera','Segunda','Tercera')

training <- trainingMod
testing <- testMod


 train_control <- trainControl(method = 'cv', number = 10, classProbs = TRUE, summaryFunction = twoClassSummary)

#Neural Networks
nnGrid <- expand.grid(.size = c(1,2,3,4,5,6,7),
                      .decay = c(0, .01, .1, .2, .3, .4, .5, 1, 2))

nnModel <- caret::train(Survived ~ FareIndividualBySex+IsMother+LoneWolfs+TicketFreq+ageGroupsByCut+Embarked+Pclass
                method = 'nnet', tuneGrid = nnGrid,
                metric = 'ROC', trace = FALSE)

pcaNNModel <- caret::train(Survived ~ ., data = training, trControl = train_control,
                method = 'pcaNNet', tuneGrid = nnGrid,
                metric = 'ROC', trace = FALSE)

#models <- list(nn = nnModel, pcann = pcaNNModel)
# models_preds <- lapply(models, predict, newdata = testing, type = 'prob')
# models_probs <- as.data.frame(sapply(models_preds, function(df){1 - df$N}))
# models_probs$Survived <- training$Survived
```

```r
nnModelTest <- data.frame(PassengerId = testing$PassengerId,
                          Survived = ifelse(predict(nnModel, testing) == 'Si',1,0))
pcaNNModelTest <- data.frame(PassengerId = 892:1309,
                          Survived = ifelse(predict(pcaNNModel, testing) == 'Si',1,0))

write.csv(nnModelTest, 'Pred_nnModel.csv',row.names = FALSE)
write.csv(pcaNNModelTest, 'Pred_pcaNNModel.csv',row.names = FALSE)
```

5. Representación de los resultados a partir de tablas y gráficas.

No se nos ocurre como representar los resultados obtenidos, ya que no sabemos nada más que el porcentaje de acierto que nos facilita Kaggle. Hemos usado las gráficas en las diferentes fases del proceso.

6. Resolución del problema. A partir de los resultados obtenidos, ¿cuáles son las conclusiones? ¿Los resultados permiten responder al problema?

Las conclusiones obtenidas son que los procesos automáticos para selección de variables no son tan buenos como pensábamos, puede ser que por no tener experiencia con ellos estemos haciendo algo mal o simplemente que las variables que obtuvimos no son lo suficientemente explicativas para obtener mejores resultados.

Sobra decir que por toda la literatura que hay acerca de esta competición, hemos visto métodos de predicción que obtienen mejores resultados que los nuestros, pero decidimos probar por nuestra cuenta para ver hasta donde podíamos llegar. Inicialmente nos ha costado que todo simplemente funcionase, pero hemos aprendido bastante y poco a poco hemos mejorado nuestros conocimientos de las técnicas que hemos usado, lo que se ha ido reflejando en una mejoría en el porcentaje de éxito. Si dispusiesemos de algo más de tiempo supongo que podríamos haber superado la barrera del 80%, probablemente lo hagamos próximamente, pero ya no será como parte de la práctica.